



УКРАЇНА

(19) **UA** (11) **115186** (13) **C2**
(51) МПК**H03M 7/42** (2006.01)**H04N 7/52** (2011.01)**H04N 7/24** (2011.01)МІНІСТЕРСТВО
ЕКОНОМІЧНОГО
РОЗВИТКУ І ТОРГІВЛІ
УКРАЇНИ**(12) ОПИС ДО ПАТЕНТУ НА ВІНАХІД**

(21) Номер заявки:	а 2016 00953	(72) Винахідник(и):	Георге Валері (DE), Бросс Бенджамін (DE), Кірххоффер Хайнер (DE), Марпе Детлеф (DE), Нгуєн Тунг (DE), Прайсс Маттіас (DE), Зікманн Міша (DE), Штегеманн Ян (DE), Віганд Томас (DE)
(22) Дата подання заявки:	18.06.2012	(73) Власник(и):	ДЖ.І. ВІДІЕУ КЕМПРЕШН, ЛЛСІ, 8 Southwoods Boulevard, Albany, New York 12211, USA (US)
(24) Дата, з якої є чинними права на винахід:	25.09.2017	(74) Представник:	Пахаренко Антоніна Павлівна, реєстр. №4
(31) Номер попередньої заявки відповідно до Паризької конвенції:	61/497,794, 61/508,506	(56) Перелік документів, взятих до уваги експертизою:	US 5381145 A, 10.01.1995 US 5717394 A, 10.02.1998 US 6900748 B2, 31.05.2005 US 6943710 B2, 13.09.2005 US 5847776 A, 08.12.1998 WO 2009017301 A1, 05.02.2009
(32) Дата подання попередньої заявки відповідно до Паризької конвенції:	16.06.2011, 15.07.2011		
(33) Код держави-учасниці Паризької конвенції, до якої подано попередню заявку:	US, US		
(41) Публікація відомостей про заявку:	25.07.2016, Бюл.№ 14		
(46) Публікація відомостей про видачу патенту:	25.09.2017, Бюл.№ 18		
(62) Номер та дата подання попередньої заявки, з якої виділено заявку, позначену кодом (21):	, а201314707, 18.06.2012		

(54) ЕНТРОПІЙНЕ КОДУВАННЯ РІЗНИЦЬ ВЕКТОРІВ РУХУ**(57) Реферат:**

Описується декодер для декодування відеоданих з потоку даних, у який кодуються горизонтальні і вертикальні компоненти різниць векторів руху, з використанням бінаризацій горизонтальних і вертикальних компонентів, при цьому бінаризації зрівнюють зрізаний унарний код горизонтальних і вертикальних компонентів, відповідно, в першому інтервалі області горизонтальних і вертикальних компонентів нижче критичної величини, і комбінацію префіксу у формі зрізаного унарного коду для критичної величини і суфікса у формі експоненціального коду Голомба горизонтальних і вертикальних компонентів, відповідно, в другому інтервалі області горизонтальних і вертикальних компонентів включно і з перевищенням критичної величини, при цьому критична величина дорівнює двійці, а експоненціальний код Голомба має порядок одиниця. Ентропійний декодер сконфігурований для одержання для горизонтальних і вертикальних компонентів різниць векторів руху зрізаного унарного коду з потоку даних з використанням адаптивного до контексту бінарного ентропійного декодування з точно одним контекстом на положення інформаційної величини зрізаного унарного коду, який є спільним для горизонтальних і вертикальних компонентів різниць векторів руху, і експоненціального коду Голомба з використанням сталого рівномірного байпасного режиму для одержання результатів бінаризацій різниць векторів руху. Десимволізатор сконфігурований для

UA 115186 C2

дебінаризації результатів бінаризації синтаксичних елементів різниці векторів руху для одержання цілих величин горизонтальних і вертикальних компонентів різниць векторів руху. Реконструктор сконфігурований для відновлення відеоданих на основі цілих величин горизонтальних і вертикальних компонентів різниць векторів руху.

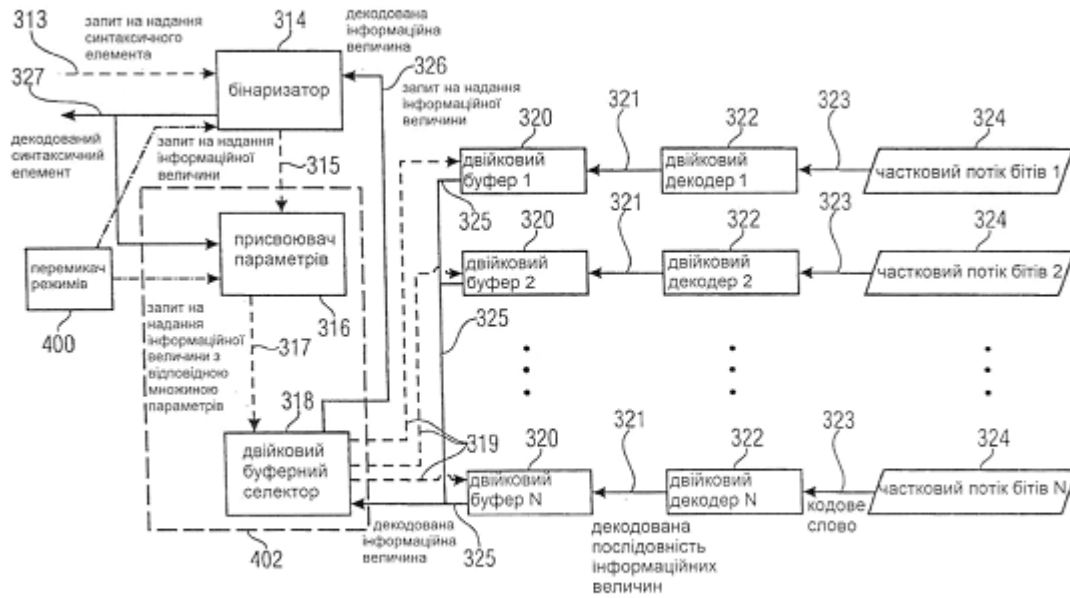


Fig. 8

Представлений винахід стосується концепції ентропійного кодування для кодування відеоданих.

В рівні техніки відомо багато відеокодеків. Загалом, ці кодеки зменшують кількість даних, необхідних для представлення відеоконтенту, тобто, вони стискають дані. В контексті кодування відеоданих, відомо, що стискання відеоданих переважно досягається послідовним застосуванням різних технологій кодування: для прогнозування змісту картинки використовується прогнозування з компенсацією руху. Вектори руху, визначені в прогнозуванні з компенсацією руху, а також залишок прогнозування піддаються ентропійному кодуванню без втрат. Для подальшого зменшення кількості даних, самі по собі вектори руху піддаються прогнозуванню так, що просто різниці векторів руху, які представляють їх залишок, повинні ентропійно кодуватися. В стандарті H.264, наприклад, тільки що описана процедура застосовується для передачі інформації про різниці векторів руху. Зокрема, різниці векторів руху бінаризуються з одержанням рядків інформаційних величин, які відповідають комбінації зрізаного унарного коду і, від певної критичної величини, експоненціального коду Голомба. Хоча інформаційні величини експоненціального коду Голомба легко кодуються з використанням рівномірного байпасного режиму з фіксованою ймовірністю 0,5, для перших інформаційних величин передбачаються декілька контекстів. Критична величина вибирається рівною дев'яти. Відповідно, передбачається велика кількість контекстів для кодування різниць векторів руху.

Однак, передбачення великої кількості контекстів не тільки підвищує складність кодування, але й може також негативно впливати на ефективність кодування: якщо до контексту звертаються занадто рідко, то адаптація ймовірності, тобто адаптація оцінки ймовірності, зв'язаної з відповідним контекстом, під час ентропійного кодування не є ефективною. Відповідно, непридатні застосовувані оцінки ймовірності оцінюють реальну статистику символів. Більше того, якщо для певної інформаційної величини бінаризації передбачається декілька контекстів, то вибір серед них може потребувати перевірки значень сусідніх інформаційних величин/синтаксичних елементів, необхідність в яких може перешкоджати виконанню процесу декодування. З іншого боку, якщо надається занадто мала кількість контекстів, то інформаційні величини з сильно варійованою реальною статистикою символів групуються разом в одному контексті і, відповідно, оцінка ймовірності, зв'язана з таким контекстом, ефективно не кодує зв'язані з нею інформаційні величини. Існує постійна потреба у подальшому підвищенні ефективності ентропійного кодування різниць векторів руху.

Відповідно, задачею представленого винаходу є надання такої концепції кодування.

Ця задача вирішується об'єктом доданих незалежних пунктів формули винаходу.

Основним відкриттям представленого винаходу є те, що ефективність ентропійного кодування різниць векторів руху може додатково підвищуватися шляхом зменшення критичної величини, до якої використовується зрізаний унарний код для бінаризації різниць векторів руху, до двох так, що існують просто два положення інформаційної величини зрізаного унарного коду і якщо порядок одного використовується для експоненціального коду Голомба для бінаризації різниць векторів руху від критичної величини, і якщо, окрім того, точно один контекст передбачається для двох положень інформаційної величини зрізаного унарного коду, відповідно, так, що вибір контексту на основі інформаційних величин або значень синтаксичного елемента сусідніх блоків зображення не потрібен і уникається занадто точна класифікація інформаційних величин у цих положеннях інформаційних величин відповідно до контекстів так, що адаптація ймовірності здійснюється належним чином, і якщо однакові контексти використовуються для горизонтальних і вертикальних компонентів, таким чином додатково послаблюючи негативні впливи занадто дрібного підрозбиття контексту.

Окрім того, було виявлено, що тільки що згадані параметри стосовно ентропійного кодування різниць векторів руху особливо цінні при поєднанні їх з сучасними способами прогнозування векторів руху і зменшення необхідної кількості різниць векторів руху, які передаються. Наприклад, може передбачатися багато предикторів векторів руху для одержання упорядкованого списку предикторів векторів руху і індекс у цьому списку предикторів векторів руху може використовуватися для визначення реального предиктора вектора руху, залишок прогнозування якого представляється розглядуваною різницею векторів руху. Хоча інформація про використовуваний індекс списку, повинна одержуватися з потоку даних на декодувальній стороні, загальна якість прогнозування векторів руху підвищується і, відповідно, величина різниць векторів руху додатково зменшується так, що, в цілому, ефективність кодування додатково збільшується і зменшення критичної величини та спільне використання контексту для горизонтальних і вертикальних компонентів різниць векторів руху узгоджуються з таким покращеним прогнозуванням векторів руху. З іншого боку, може застосовуватися злиття для зменшення кількості різниць векторів руху, які передаються в потоці даних: для цього,

інформація про злиття може передаватися в потоці даних, який сигналізує декодувальним блокам про підрозбиття блоків, які групуються в групу блоків. Різниці векторів руху можуть потім передаватися в потоці даних в елементах цих об'єднаних груп замість окремих блоків, таким чином зменшуючи кількість різниць векторів руху, які повинні передаватися. Оскільки ця кластеризація блоків послаблює інтеркореляцію між різницями сусідніх векторів руху, то тільки що описана відмова від надання декількох контекстів для одного положення інформаційної величини перешкоджає схемі ентропійного кодування проводити занадто точну класифікацію згідно з контекстами в залежності від різниць сусідніх векторів руху. Скоріше концепція злиття вже використовує інтеркореляцію між різницями векторів руху сусідніх блоків і, відповідно, один контекст для одного положення інформаційної величини: однаковий контекст достатній для горизонтальних і вертикальних компонентів.

Переважні варіанти виконання даного винаходу описуються далі з посиланням на креслення, на яких:

Фіг. 1 зображує блок-схему кодера згідно з варіантом виконання;

Фіг. 2a-2c схематично зображають різні підрозбиття масиву зразків, такого як підрозбиття картинки на блоки;

Фіг. 3 зображує блок-схему декодера згідно з варіантом виконання;

Фіг. 4 детальніше зображує блок-схему кодера згідно з варіантом виконання;

Фіг. 5 детальніше зображує блок-схему декодера згідно з варіантом виконання;

Фіг. 6 схематично зображує перетворення блока з просторової області в спектральну область, одержуваний блок перетворення і його повторне перетворення;

Фіг. 7 зображує блок-схему кодера згідно з варіантом виконання;

Фіг. 8 зображує блок-схему декодера, придатного для декодування потоку бітів, генерованого кодером з Фіг. 8, згідно з варіантом виконання;

Фіг. 9 зображує схематичну діаграму, яка показує пакет даних з мультиплексованими частковими потоками бітів згідно з варіантом виконання;

Фіг. 10 зображує схематичну діаграму, яка показує пакет даних з альтернативною сегментацією, яка використовує сегменти фіксованого розміру, згідно з подальшим варіантом виконання;

Фіг. 11 зображує декодер, який використовує режим перемикавання, згідно з варіантом виконання;

Фіг. 12 зображує декодер, який використовує режим перемикавання, згідно з подальшим варіантом виконання;

Фіг. 13 зображує кодер, який підключається до декодера з Фігури. 11, згідно з варіантом виконання;

Фіг. 14 зображує кодер, який підключається до декодера з Фіг. 12, згідно з варіантом виконання;

Фіг. 15 зображує графічну залежність $pStateCtx$ і $fullCtxState/256$;

Фіг. 16 зображує декодер згідно з варіантом виконання представленого винаходу; і

Фіг. 17 зображує кодер згідно з варіантом виконання представленого винаходу.

Фіг. 18 схематично зображує бінаризацію різниці векторів руху у відповідності з варіантом виконання представленого винаходу.

Фіг. 19 схематично зображує концепцію злиття у відповідності з варіантом виконання; і

Фіг. 20 схематично зображує схему прогнозування векторів руху у відповідності з варіантом виконання.

Відзначається, що, під час опису фігур, елементи, які з'являються на декількох з цих Фігур, позначені однаковим позиційним позначенням на кожній з них і повторний опис цих елементів настільки, наскільки це стосується функцій, уникається для уникнення непотрібних повторів. Тим не менше, функції і описи, надані для однієї фігури, повинні також застосовуватися до інших Фігур, якщо чітко не вказано протилежне.

Далі, по-перше, описуються варіанти виконання загальної концепції кодування відеоданих стосовно Фіг. 1-10. Фіг. 1-6 належать до частини відеокодека, яка працює на синтаксичному рівні. Наступні фігури 8-10 належать до варіантів виконання для частини коду, яка належить до перетворення потоку синтаксичних елементів на потік даних і навпаки. Потім, спеціальні аспекти і варіанти виконання представленого винаходу описуються у формі можливих втілень загальної концепції, описаної щодо Фіг. 1-10.

Фіг. 1 зображує приклад для кодера 10, у якому можуть втілюватися аспекти представленого винаходу.

Кодер кодує масив інформаційних зразків 20 з одержанням потоку даних. Масив інформаційних зразків може представляти інформаційні зразки, які відповідають, наприклад,

значенням світла, значенням кольору, значенням яскравості, значенням колірності або подібному. Однак, інформаційні зразки можуть також бути значеннями глибини у випадку масиву зразків 20, який є картою глибини, згенерованою, наприклад, часом датчика світла або подібним.

5 Кодер 10 є блочним кодером. Тобто, кодер 10 кодує масив зразків 20 з одержанням потоку даних 30 у формі елементів з блоків 40. Кодування в елементах з блоків 40 необов'язково означає, що кодер 10 кодує ці блоки 40 цілком незалежно один від іншого. Скоріше, кодер 10 може використовувати відновлення попередньо кодованих блоків для екстраполяції або інтрапрогнозування решти блоків і може використовувати гранулярність блоків для визначення параметрів кодування, тобто, для визначення способу кодування кожної ділянки масиву зразків, яка відповідає відповідному блоку.

10 Окрім того, кодер 10 є перетворювальним кодером. Тобто, кодер 10 кодує блоки 40 шляхом використання перетворення для перенесення зразків інформації в кожному блоці 40 з просторової області на спектральну область. Може використовуватися двовимірне перетворення, таке як DCT (дискретне косинусне перетворення) FFT (швидке перетворення Фур'є) або подібне. Переважно, блоки 40 мають квадратну форму або прямокутну форму.

15 Підрозбиття масиву зразків 20 на блоки 40, зображене на Фіг. 1, просто служить для ілюстраційних цілей. Фіг. 1 зображує масив зразків 20, підрозбитий з одержанням стандартного двовимірного розташування квадратних або прямокутних блоків 40, які примикають один до іншого без налягання. Розмір блоків 40 може попередньо встановлюватися. Тобто, кодер 10 може не переносити інформацію про розмір блоків 40 в потоці даних 30 до декодувальної сторони. Наприклад, декодер може очікувати на наперед встановлений розмір блока.

20 Однак, можливі декілька альтернатив. Наприклад, блоки можуть налягати один на інший. Налягання може, однак, обмежуватися до такої міри, що кожен блок має частину, на яку не налягає будь-який сусідній блок, або так, що на кожен зразок блоків налягає максимум один блок серед сусідніх блоків, розташованих суміжно з поточним блоком вздовж наперед встановленого напрямку. Останнє повинно означати, що ліві і праві сусідні блоки можуть налягати на поточний блок для повного покриття поточного блока, але вони можуть не налягати один на інший і те ж саме застосовується для сусідніх блоків у вертикальному і діагональному напрямі.

30 Як подальша альтернатива, підрозбиття масиву зразків 20 з одержанням блоків 40 може адаптуватися до контенту масиву зразків 20 кодером 10 за допомогою інформації про використовуване підрозбиття, яка переноситься до декодера потоком бітів 30.

35 Фігури 2a-2c зображують різні приклади для підрозбиття масиву зразків 20 на блоки 40. Фіг. 2a зображує підрозбиття на основі квадрадерева масиву зразків 20 на блоки 40 різних розмірів, при цьому відповідні блоки позначені позиціями 40a, 40b, 40c і 40d із зростаючим розміром. У відповідності з підрозбиттям з Фіг. 2a масив зразків 20 спершу ділиться на стандартну двовимірну схему деревних блоків 40d, які, у свою чергу, мають індивідуальну інформацію про підрозбиття, пов'язану з нею, згідно з якою, певний деревний блок 40d може додатково підрозбиватися згідно зі структурою квадрадерева або ні. Деревний блок зліва від блока 40d ілюстративно підрозбитий на менші блоки у відповідності зі структурою квадрадерева. Кодер 10 може виконувати одне двовимірне перетворення для кожного з блоків, зображених суцільною і пунктирною лініями на Фіг. 2a. Іншими словами, кодер 10 може перетворювати масив 20 в елементи підрозбиття блока.

45 Замість підрозбиття на основі квадрадерева може використовуватися більш загальне підрозбиття на основі мультидерева і кількість дочірніх вузлів на ієрархічному рівні може відрізнятися в різних ієрархічних рівнях.

50 Фіг. 2b зображує інший приклад для підрозбиття. У відповідності з Фіг. 2b масив зразків 20 спершу ділиться на макроблоки 40b, розташовані згідно зі стандартною двовимірною схемою без взаємного налягання з упиранням один в інший, де кожен макроблок 40b зв'язаний з інформацією про підрозбиття, згідно з якою макроблок не підрозбивається або, якщо підрозбивається, то підрозбивається стандартним двовимірним способом на однакові субблоки для досягання різних гранулярностей підрозбиття для різних макроблоків. Результатом є підрозбиття масиву зразків 20 на різного розміру блоки 40 з представниками різних розмірів, які вказані позиціями 40a, 40b and 40a'. Як і на Фіг. 2a кодер 10 виконує двовимірне перетворення на кожному з блоків, зображених на Фіг. 2b суцільними і пунктирними лініями. Фіг. 2c буде обговорюватися пізніше.

60 Фіг. 3 зображує декодер 50, який здатен декодувати потік даних 30, генерований кодером 10, для відновлення відновленого варіанта 60 масиву зразків 20. Декодер 50 одержує з потоку даних 30 блок коефіцієнтів перетворення для кожного з блоків 40 і відновлює відновлений

варіант 60 шляхом виконання оберненого перетворення на кожному з блоків коефіцієнтів перетворення.

Кодер 10 і декодер 50 можуть конфігуруватися для виконання ентропійного кодування/декодування для введення інформації про блоки коефіцієнтів перетворення і, відповідно, одержання цієї інформації з потоку даних. Деталі з цього приводу у відповідності з різними варіантами виконання описуються пізніше. Слід відзначити, що потік даних 30 необов'язково містить інформацію про блоки коефіцієнтів перетворення для усіх блоків 40 масиву зразків 20. Скоріше, підмножина блоків 40 може кодуватися іншим способом з одержанням потоку бітів 30. Наприклад, кодер 10 може приймати рішення утриматися від введення блока коефіцієнтів перетворення для певного блока з блоків 40 із введенням в потік бітів 30 альтернативних параметрів кодування, замість чого надавати можливість декодеру 50 прогнозувати або, інакше, поміщати відповідний блок у відновлений варіант 60. Наприклад, кодер 10 може виконувати аналіз текстури для поміщення блоків в масиві зразків 20, який може заповнюватися на декодувальній стороні декодером у вигляді синтезу текстури і вказує це, відповідним чином, в потоці бітів.

Як це обговорюється стосовно наступних фігур, блоки коефіцієнтів перетворення необов'язково представляють спектральне представлення області зразків первинної інформації відповідного блока 40 масиву зразків 20. Скоріше такий блок коефіцієнтів перетворення може представляти спектральне представлення області залишкових величин прогнозування відповідного блока 40. Фіг. 4 зображує варіант виконання для такого кодера. Кодер з Фіг. 4 містить фазу перетворення 100, ентропійний кодер 102, фазу оберненого перетворення 104, предиктор 106 і блок віднімання 108, а також суматор 110. Блок віднімання 108, фаза перетворення 100 і ентропійний кодер 102 послідовно з'єднані в порядку, згаданому між входом 112 і виходом 114 кодера з Фіг. 4. Фаза оберненого перетворення 104, суматор 110 і предиктор 106 з'єднані в порядку, згаданому між виходом фази перетворення 100 і інверсним входом блоку віднімання 108, при цьому вихід предиктора 106 також з'єднаний з подальшим входом суматора 110.

Кодер з Фіг. 4 зображає прогнозувальний перетворювальний блочний кодер. Тобто, блоки масиву зразків 20, який надходить на вхід 112, прогнозуються з попередньо кодованих і відновлених частин того ж масиву зразків 20 або попередньо кодованих і відновлених інших масивів зразків, які можуть передувати або слідувати за поточним масивом зразків 20 в даному часі. Прогнозування виконується предиктором 106. Блок віднімання 108 віднімає дані прогнозу від такого первинного блока, а фаза перетворення 100 виконує двовимірне перетворення на залишкових величинах прогнозування. Двовимірне перетворення саме по собі або наступний захід всередині фази перетворення 100 може приводити до дискретизації коефіцієнтів перетворення в блоках коефіцієнтів перетворення. Дискретизовані блоки коефіцієнтів перетворення кодуються без втрат, наприклад, ентропійним кодуванням в ентропійному кодері 102 з виходом одержуваного потоку даних на виході 114. Фаза оберненого перетворення 104 відновлює дискретизований залишок, а суматор 110, у свою чергу, поєднує відновлений залишок відповідними даними прогнозування для одержання відновлених зразків інформації, на основі яких предиктор 106 може прогнозувати вищезгадані на даний момент кодовані прогнозувальні блоки. Предиктор 106 може використовувати різні режими прогнозування, такі як режими інтрапрогнозування і режими інтерпрогнозування, для прогнозування блоків, а параметри прогнозування направляються до ентропійного кодера 102 для введення в потік даних. Для кожного інтерпрогнозованого прогнозувального блока відповідні дані руху вводяться в потік бітів за допомогою ентропійного кодера 114 для надання можливості декодувальній стороні повторно робити прогноз. Дані руху для прогнозувального блока картинки можуть використовувати синтаксичну частину, яка містить синтаксичний елемент, який представляє різницю векторів руху, яка диференціально кодує вектор руху для поточного прогнозувального блока відносно предиктора вектора руху, одержаного, наприклад, описаним способом з векторів руху сусідніх вже кодованих прогнозувальних блоків.

Тобто, у відповідності з варіантом виконання з Фіг. 4, блоки коефіцієнтів перетворення скоріше представляють спектральне представлення залишку масиву зразків, а ніж його інформаційні зразки. Тобто, у відповідності з варіантом виконання з Фіг. 4, послідовність синтаксичних елементів може надходити в ентропійний кодер 102 для ентропійного кодування з одержанням потоку даних 114. Послідовність синтаксичних елементів може містити синтаксичні елементи різниці векторів руху для інтерпрогнозувальних блоків і синтаксичні елементи, які стосуються карти значущості, яка вказує положення рівнів значущості коефіцієнта перетворення, а також синтаксичні елементи, які самі по собі визначають рівні значущості коефіцієнта перетворення для блоків перетворення.

Слід відзначити, що для варіанта виконання з Фіг. 4 існують декілька альтернатив, деякі з яких описані у вступній частині опису, який включений в опис Фіг. 4.

Фіг. 5 зображає декодер, здатний декодувати потік даних, генерований кодером з Фіг. 4. Декодер з Фіг. 5 містить ентропійний декодер 150, фазу оберненого перетворення 152, суматор 154 і предиктор 156. Ентропійний декодер 150, фаза оберненого перетворення 152 і суматор 154 послідовно з'єднані між входом 158 і виходом 160 декодера з Фіг. 5 у згаданому порядку. Подальший вихід ентропійного декодера 150 з'єднаний з предиктором 156, який, у свою чергу, з'єднаний між виходом суматора 154 та його подальшим входом. Ентропійний декодер 150 виділяє з потоку даних, який надходить в декодер з Фіг. 5 на вхід 158, блоки коефіцієнтів перетворення, де для блоків коефіцієнтів перетворення на фазі 152 застосовується обернене перетворення для одержання залишкового сигналу. Залишковий сигнал поєднується з прогнозом від предиктора 156 на суматорі 154 для одержання відновленого блока відновленого варіанта масиву зразків на виході 160. На основі відновлених варіантів предиктор 156 генерує прогнози, таким чином повторно створюючи прогнози, виконані предиктором 106 на кодувальній стороні. Для одержання тих же прогнозів, що використовуються на кодувальній стороні, предиктор 156 використовує параметри прогнозування, які ентропійний декодер 150 також одержує з потоку даних на вході 158.

Слід відзначити, що у вищеописаних варіантах виконання просторова гранулярність, з якою виконується прогнозування і перетворення залишку, не повинна бути рівною між собою. Це зображено на Фіг. 2С. Ця фігура зображує підрозбиття для прогнозувальних блоків прогнозувальної гранулярності суцільними лініями і залишкову гранулярність -пунктирними лініями. Як можна побачити, підрозбиття можуть вибиратися кодером незалежно одне від іншого. Для більшої точності, синтаксис потоку даних може передбачати визначення підрозбиття залишку незалежно від підрозбиття прогнозу. Альтернативно, підрозбиття залишку може бути продовженням підрозбиття прогнозу так, що кожен залишковий блок або дорівнює або є належною підмножиною прогнозувального блока. Це зображено на Фіг. 2а і Фіг. 2b, наприклад, де знову прогнозувальна гранулярність зображена суцільними лініями, а залишкова гранулярність - пунктирними лініями. Тобто, на Фіг. 2а-2с усі блоки, які мають позиційне позначення, повинні бути залишковими блоками, для яких повинно виконуватися двовимірне перетворення, тоді як блоки, вказані товщими суцільними лініями і охоплюють блоки 40а, вказані пунктирними лініями, наприклад, повинні бути прогнозувальними блоками, для яких індивідуально виконується визначення параметрів прогнозування.

Вищезгадані варіанти виконання повинні перетворювати загалом такий блок (залишковий або первинний) зразків на кодувальній стороні з одержанням блока коефіцієнтів перетворення, який, у свою чергу, повинен інверсно перетворюватися на відновлений блок зразків на декодувальній стороні. Це зображено на Фіг. 6. Фіг. 6 зображає блок зразків 200. У випадку Фіг. 6, цей блок 200 є ілюстративно квадратним і за розміром становить 4×4 зразки 202. Зразки 202 стандартно розташовані вздовж горизонтального напрямку x і вертикального напрямку y . Завдяки вищезгаданому двовимірному перетворенню Т блок 200 перетворюється на спектральну область, зокрема на блок 204 коефіцієнтів 206 перетворення, який має той же розмір що й блок 200. Тобто, блок перетворення 204 має стільки коефіцієнтів 206 перетворення, скільки блок 200 має зразків як в горизонтальному напрямі, так і у вертикальному напрямі. Однак, оскільки перетворення Т є спектральним перетворенням, то положення коефіцієнтів 206 перетворення в блоці перетворення 204 не відповідають просторовим положенням, а скоріше спектральним компонентам контенту блока 200. Зокрема, горизонтальна вісь блока перетворення 204 відповідає осі, вздовж якої спектральна частота в горизонтальному напрямі монотонно зростає, тоді як вертикальна вісь відповідає осі, вздовж якої просторова частота у вертикальному напрямі монотонно зростає, де коефіцієнт перетворення компонента DC розташований в куті - тут ілюстративно верхній лівий кут - блока 204 так, що в нижньому правому куті розташований коефіцієнт 206 перетворення, який відповідає найвищій частоті як в горизонтальному, так і у вертикальному напрямі. Нехтуючи просторовим напрямом, просторова частота, якій належить певний коефіцієнт 206 перетворення, головним чином зростає від верхнього лівого кута до нижнього правого кута. Завдяки оберненому перетворенню T^{-1} блок перетворення 204 повторно переноситься зі спектральної області на просторову область для повторного одержання копії 208 блока 200. У випадку відсутності дискретизації/втрати під час перетворення, відновлення повинно бути досконалим.

Як вже відзначалося вище, з Фіг. 6 можна побачити, що блок 200 більшого розміру збільшує спектральну роздільну здатність одержуваного спектрального представлення 204. З іншого боку, шум дискретизації має тенденцію до поширення по усьому блоку 208 і, таким чином, круті і дуже локалізовані об'єкти в блоках 200 мають тенденцію до появи відхилень повторно

перетвореного блока відносно первинного блока 200 завдяки шуму дискретизації. Основною перевагою використання більших блоків є, однак, те, що відношення між кількістю значущих, тобто, ненульових (дискретизованих) коефіцієнтів перетворення, тобто рівнів, з одного боку, і кількістю несуттєвих коефіцієнтів перетворення, з іншого боку, може зменшуватися в більших
 5 блоках порівняно з меншими блоками, таким чином забезпечуючи кращу ефективність кодування. Іншими словами, часто, рівні значущості коефіцієнта перетворення, тобто, коефіцієнти перетворення, не дискретизовані в нуль, рідко розподіляються по блоку перетворення 204. Завдяки цьому, у відповідності з варіантами виконання, описаними детальніше нижче, положення рівнів значущості коефіцієнта перетворення сигналізуються в
 10 потоці даних у вигляді карти значущості. Окремо від них величини значущого коефіцієнта перетворення, тобто, рівні значущості коефіцієнта перетворення у випадку коефіцієнтів перетворення, які дискретизуються, переносяться в потоці даних.

Усі описані вище кодери і декодери, таким чином, конфігуруються для певного синтаксису синтаксичних елементів. Тобто, вищезгадані синтаксичні елементи, такі як рівні значущості коефіцієнта перетворення, синтаксичні елементи, які стосуються карти значущості блоків перетворення, синтаксичні елементи даних руху, які стосуються інтерпрогнозувальних блоків і так далі, припускаються тими, що послідовно розташовані в потоці даних попередньо встановленим способом. Такий попередньо встановлений спосіб може представлятися у формі псевдокоду, як це робиться, наприклад, в стандарті H.264 або інших відеокодеках.

Ще іншими словами, вищенаведений опис головним чином має справу з перетворенням мультимедійних даних, тут ілюстративних відеоданих, на послідовність синтаксичних елементів у відповідності з попередньо визначеною синтаксичною структурою, яка описує певні типи синтаксичних елементів, їх семантику і порядок серед них. Ентропійний кодер і ентропійний декодер з Фіг. 4 і 5 можуть конфігуруватися для роботи і можуть структуруватися, як вказано
 20 далі. Те ж саме відповідає за виконання перетворення між послідовністю синтаксичних елементів і потоком даних, тобто, символьним або бітовим потоком.

Ентропійний кодер згідно з варіантом виконання зображений на Фіг. 7. Кодер без втрат перетворює потік синтаксичних елементів 301 на множину з двох або більшої кількості часткових бітових потоків 312.

В переважному варіанті виконання винаходу кожен синтаксичний елемент 301 зв'язується з категорією множини з однієї або більшої кількості категорій, тобто, типом синтаксичного елемента. Як приклад, категорії можуть специфікувати тип синтаксичного елемента. В контексті гібридного кодування відеоданих окрема категорія може зв'язуватися з режимами макроблочного кодування, режимами блочного кодування, індексами еталонної картинки, різницями векторів руху, ідентифікаторами підрозбиття, ідентифікаторами кодованих блоків, параметрами дискретизації, рівнями значущості коефіцієнта перетворення і так далі. В інших областях застосування, таких як аудіо-кодування, кодування мови, кодування тексту, кодування документа або кодування загальних даних, можливі різні категоризації синтаксичних елементів.

Загалом, кожен синтаксичний елемент може приймати величину із скінченної або зчисленної нескінченної множини величин, де множина можливих величин синтаксичних елементів може відрізнятися для різних категорій синтаксичних елементів. Наприклад, існують бінарні синтаксичні елементи, а також цілі синтаксичні елементи.

Для зниження складності алгоритму кодування і декодування, і для надання можливості виконання загальної схеми кодування і декодування для різних синтаксичних елементів і категорій синтаксичних елементів, синтаксичні елементи 301 перетворюються на упорядковані множини бінарних рішень і ці бінарні рішення потім обробляються простими алгоритмами бінарного кодування. Тому, бінаризатор 302 бісктивно перетворює величину кожного синтаксичного елемента 301 на послідовність (або рядок або слово) з двійкових кодів 303. Послідовність двійкових кодів 303 представляє множину упорядкованих бінарних рішень.
 50 Кожний двійковий код 303 або бінарне рішення може приймати одну величину з множини з двох величин, наприклад одну з величин, вибрану з 0 і 1. Схема бінаризації може відрізнятися для різних категорій синтаксичних елементів. Схема бінаризації для окремої категорії синтаксичних елементів може залежати від множини можливих величин синтаксичних елементів і/або інших властивостей синтаксичного елемента для окремої категорії.

Таблиця 1 зображує три ілюстративні схеми бінаризації для зчисленних нескінченних множин. Схеми бінаризації для зчисленних нескінченних множин можуть також застосовуватися для скінчених множин величин синтаксичних елементів. Зокрема, для великих скінчених множин величин синтаксичних елементів неефективність (одержувана з невикористаних послідовностей двійкових кодів) може бути нехтуваною, але універсальність таких схем бінаризації надає перевагу з точки зору складності та вимог до пам'яті. Для малих скінчених
 60

множин величин синтаксичних елементів часто вигідно (з точки зору ефективності кодування) адаптувати схему бінарizaції до кількості можливих символічних величин.

Таблиця 2 зображає три ілюстративні схеми бінарizaції для скінченних множин з 8 величин. Схеми бінарizaції для скінчених множин можна одержати з універсальних схем бінарizaції для зчисленних нескінченних множин шляхом модифікації деяких послідовностей двійкових кодів у такий спосіб, що скінчені множини послідовностей двійкових кодів представляють код без резервування (і який потенційно повторно упорядковує послідовності двійкових кодів). Як приклад, схема бінарizaції з використанням зрізаних унарних кодів в Таблиці 2 була створена шляхом модифікації послідовності двійкових кодів для синтаксичного елемента 7 бінарizaції з використанням універсальних унарних кодів (дивіться Таблицю 1). Переупорядкована бінарizaція з використанням експоненціальних кодів Голомба порядку 0 в Таблиці 2 створена модифікацією послідовності двійкових кодів для синтаксичного елемента 7 бінарizaції з використанням універсальних експоненціальних кодів Голомба порядку 0 (дивіться Таблицю 1) і повторним упорядкуванням послідовностей двійкових кодів (послідовність зрізаних двійкових кодів для символу 7 присвоєна символу 1). Для скінченних множин синтаксичних елементів також можна використовувати несистематичні/неуніверсальні схеми бінарizaції, як ілюструється в останньому стовпчику Таблиці 2.

Таблиця 1:

Приклади бінарizaції для зчисленних нескінченних множин (або великих скінченних множин)

Символьна величина	Бінарizaція з використанням унарних кодів	Бінарizaція з використанням експоненціальних кодів Голомба порядку 0	Бінарizaція з використанням експоненціальних кодів Голомба порядку 1
0	1	1	10
1	01	010	11
2	001	011	0100
3	0001	0010 0	0101
4	0000 1	0010 1	0110
5	0000 01	0011 0	0111
6	0000 001	0011 1	0010 00
7	0000 0001	0001 000	0010 01

Таблиця 2:

Приклади бінарizaції для скінченних множин

Символьна величина	Бінарizaція з використанням зрізаного унарного коду	переупорядкована бінарizaція з використанням експоненціальних кодів Голомба порядку 0	Несистематична бінарizaція
0	1	1	000
1	01	000	001
2	001	010	01
3	0001	011	1000
4	0000 1	0010 0	1001
5	0000 01	0010 1	1010
6	0000 001	0011 0	1011 0
7	0000 000	0011 1	1011 1

Кожен двійковий код 303 послідовності двійкових кодів, створеної бінаризатором 302, подається в присвоювач 304 параметрів у послідовному порядку. Присвоювач параметрів присвоює множину з одного або більшої кількості параметрів кожному двійковому коду 303 і видає двійковий код з відповідною множиною параметрів 305.

Множина параметрів визначається точно одним і тим же способом в кодері і декодері. Множина параметрів може складатися з одного або більшої кількості наступних параметрів:

Зокрема, присвоювач 304 параметрів може конфігуруватися для присвоювання поточному двійковому коду 303 моделі контексту. Наприклад, присвоювач 304 параметрів може вибрати один з доступних індексів контексту для поточного двійкового коду 303. Доступна множина контекстів для поточного двійкового коду 303 може залежати від типу двійкового коду, який, у

свою чергу, може визначатися типом/категорією синтаксичного елемента 301, в результаті бінаризації якого поточний двійковий код 303 є його частиною, а положення поточного двійкового коду 303 знаходиться в пізнішій бінаризації. Вибір контексту серед множини доступних контекстів може залежати від попередніх двійкових кодів і синтаксичних елементів, пов'язаних з ними. Кожен з цих контекстів має модель ймовірності, зв'язану з ним, тобто, засіб для оцінки ймовірності для поточної інформаційної величини набуття одного з двох можливих значень інформаційної величини. Модель ймовірності може, зокрема, бути засобом для оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення, при цьому модель ймовірності додатково визначається ідентифікатором, який специфікує оцінку, яке з двох можливих значень інформаційної величини представляє її менш ймовірне або більш ймовірне значення поточного двійкового коду 303. У випадку просто одного контексту, який доступний для поточної інформаційної величини, вибір контексту може не виконуватися. Як буде описано детальніше нижче, присвоювач 304 параметрів може також виконувати адаптацію моделі ймовірності для адаптації моделей ймовірностей, зв'язаних з різними контекстами, до реальної статистики відповідних інформаційних величин, які належать відповідним контекстам.

Як буде також детальніше описуватися нижче, присвоювач 304 параметрів може працювати диференційно в залежності від вискоєфективного режиму (HE) або режиму низької складності (LC), який активується. В обох режимах модель ймовірності зв'язує поточну інформаційну величину двійкового коду 303 з будь-яким з двійкових кодерів 310, як буде описано нижче, а режим роботи присвоювача 304 параметрів має тенденцію бути менш складним в режимі LC, однак, ефективність кодування зростає у вискоєфективному режимі внаслідок того, що присвоювач 304 параметрів, який зв'язує окремі інформаційні величини двійкового коду 303 з окремими кодерами 310, акуратніше адаптований до статистики інформаційних величин, таким чином оптимізуючи ентропію відносно режиму LC.

Кожна інформаційна величина з відповідною множиною параметрів 305, яка є виходом присвоювача 304 параметрів, подається до двійкового буферного селектора 306. Двійковий буферний селектор 306 потенційно модифікує значення вхідної інформаційної величини 305 на основі її значення і відповідних параметрів 305, і подає вихідну інформаційну величину 307 з потенційно модифікованим значенням до одного з двох або більшої кількості двійкових буферів 308. Двійковий буфер 308, до якого надсилається вихідна інформаційна величина 307, визначається на основі значення вхідної інформаційної величини 305 і/або величини відповідних параметрів 305.

В переважному варіанті виконання винаходу двійковий буферний селектор 306 не модифікує значення інформаційної величини, тобто, вихідна інформаційна величина 307 має завжди те ж значення що й вхідна інформаційна величина 305. В подальшому переважному варіанті виконання винаходу двійковий буферний селектор 306 визначає значення вихідної інформаційної величини 307 на основі значення вхідної інформаційної величини 305 і відповідне значення для оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень. В переважному варіанті виконання винаходу значення вихідної інформаційної величини 307 встановлюється рівним значенню вхідної інформаційної величини 305, якщо значення для оцінки ймовірності набуття поточною інформаційною величиною одного з двох можливих її значень менше за (або менше або рівне) конкретне порогове значення; якщо значення для оцінки ймовірності набуття поточною інформаційною величиною одного з двох можливих її значень більше або рівне (або більше) конкретного порогового значення, то значення вихідної інформаційної величини 307 модифікується (тобто, воно встановлюється рівним протилежному значенню вхідної інформаційної величини). В подальшому переважному варіанті виконання винаходу значення вхідної інформаційної величини 307 встановлюється рівним значенню вхідної інформаційної величини 305, якщо значення для оцінки ймовірності набуття поточною інформаційною величиною одного з двох можливих її значень більше за (або більше або рівне) конкретне порогове значення; якщо значення для оцінки ймовірності набуття поточною інформаційною величиною одного з двох можливих її значень менше або рівне (або менше) конкретного порогового значення, то значення вихідної інформаційної величини 307 модифікується (тобто, воно встановлюється рівним протилежному значенню вхідної інформаційної величини). В переважному варіанті виконання винаходу значення порогу відповідає величині 0,5 для оціненої ймовірності набуття обох можливих значень інформаційної величини.

В подальшому переважному варіанті виконання винаходу двійковий буферний селектор 306 визначає значення вихідної інформаційної величини 307 на основі значення вхідної інформаційної величини 305 і відповідного ідентифікатора, який специфікує оцінку, завдяки якій

одне з двох можливих значень інформаційної величини представляє менш ймовірне або більш ймовірне значення для поточної інформаційної величини. В переважному варіанті виконання винаходу значення вихідної інформаційної величини 307 встановлюється рівним значенню вхідної інформаційної величини 305, якщо ідентифікатор специфікує, що перше з двох можливих значень інформаційної величини представляє менш ймовірне (або більш ймовірне) значення для поточної інформаційної величини, а значення вихідної інформаційної величини 307 модифікується (тобто, воно встановлюється рівним протилежному значенню вхідної інформаційної величини), якщо ідентифікатор специфікує, що друге з двох можливих значень інформаційної величини представляє менш ймовірне (або більш ймовірне) значення для поточної інформаційної величини.

В переважному варіанті виконання винаходу двійковий буферний селектор 306 визначає двійковий буфер 308, до якого надсилається значення вихідної інформаційної величини 307, на основі відповідного значення для оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень. В переважному варіанті виконання винаходу множина можливих значень для визначення оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень є скінченою і двійковий буферний селектор 306 містить таблицю, яка точно зв'язує один двійковий буфер 308 з кожним можливим значенням для оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень, де різні значення для визначення оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень можуть зв'язуватися з одним і тим же двійковим буфером 308. В подальшому переважному варіанті виконання винаходу інтервал можливих значень для визначення оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень ділиться на ряд інтервалів, двійковий буферний селектор 306 визначає індекс інтервалу для поточного значення для оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень і двійковий буферний селектор 306 містить таблицю, яка точно зв'язує один двійковий буфер 308 з кожним можливим значенням для індексу інтервалу, де різні значення для індексу інтервалу можуть зв'язуватися з одним і тим же двійковим буфером 308. В переважному варіанті виконання винаходу вхідні інформаційні величини 305 з протилежними значеннями для оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень (протилежні значення є ті, які представляють оцінки ймовірності P і $1 - P$) надсилаються в один і той же двійковий буфер 308. В подальшому переважному варіанті виконання винаходу зв'язування значення для оцінки ймовірності набуття поточною інформаційною величиною одного з двох її можливих значень з окремим двійковим буфером адаптується з часом, наприклад, для гарантії того, що створені часткові потоки бітів мають подібні швидкості передачі бітів. Далі нижче, індекс інтервалу буде також називатися ріре індексом, тоді як ріре індекс разом з індексом деталізації і ідентифікатором, який вказує більш ймовірне значення інформаційної величини, індексують фактичну модель ймовірності, тобто, оцінку ймовірності.

В подальшому переважному варіанті виконання винаходу двійковий буферний селектор 306 визначає двійковий буфер 308, до якого надсилають вихідну інформаційну величину 307, на основі відповідного значення для оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення. В переважному варіанті виконання винаходу множина можливих значень для оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення є скінченою і двійковий буферний селектор 306 містить таблицю, яка точно зв'язує один двійковий буфер 308 з кожним можливим значенням оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення, де різні значення для визначення оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення можуть зв'язуватися з одним і тим же двійковим буфером 308. В подальшому переважному варіанті виконання винаходу інтервал можливих значень для визначення оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення ділиться на ряд інтервалів, двійковий буферний селектор 306 визначає індекс інтервалу для поточного визначення оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення, а двійковий буферний селектор 306 містить таблицю, яка точно зв'язує один двійковий буфер 308 з кожним можливим значенням для індексу інтервалу, де різні значення для індексу інтервалу можуть зв'язуватися з одним і тим же двійковим буфером 308. В подальшому переважному варіанті виконання винаходу зв'язування значення для оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення з окремим

двійковим буфером адаптується з часом, наприклад, для гарантії того, що створені часткові потоки бітів мають подібні швидкості передачі бітів.

Кожен з двох або більшої кількості двійкових буферів 308 з'єднаний точно з одним двійковим кодером 310 і кожен двійковий кодер з'єднаний тільки з одним двійковим буфером 308. Кожен двійковий кодер 310 зчитує інформаційні величини з відповідного двійкового буфера 308 і перетворює послідовність інформаційних величин 309 на кодове слово 311, яке представляє послідовність бітів. Двійкові буфери 308 представляють буфери, які працюють по принципу «першим надійшов-першим вийшов»; інформаційні величини, які подаються пізніше (у послідовному порядку) у двійковий буфер 308, не кодуються перед інформаційними величинами, які подаються раніше (у послідовному порядку) у двійковий буфер. Кодові слова 311, які є виходом з окремого двійкового кодера 310, записуються в окремий частковий потік бітів 312. Загальний алгоритм кодування перетворює синтаксичні елементи 301 на два або більшу кількість часткових потоків бітів 312, де кількість часткових потоків бітів дорівнює кількості двійкових буферів і двійкових кодерів. В переважному варіанті виконання винаходу двійковий кодер 310 перетворює змінну кількість інформаційних величин 309 на кодове слово 311 із змінною кількістю бітів. Однією перевагою вище-і нижчеописаних варіантів виконання винаходу є те, що кодування інформаційних величин може здійснюватися паралельно (наприклад, для різних груп значень ймовірності), що зменшує тривалість обробки для декількох втілень.

Іншою перевагою варіантів виконання винаходу є те, що кодування інформаційних величин, яке виконується двійковими кодерами 310, може спеціально розроблятися для різних множин параметрів 305. Зокрема, кодування і декодування інформаційних величин можуть оптимізуватися (з точки зору ефективності кодування і/або складності) для різних груп оцінених ймовірностей. З одного боку, це дозволяє послабити складність кодування/декодування і, з іншого боку, це дозволяє покращити ефективність кодування. В переважному варіанті виконання винаходу двійкові кодери 310 втілюють різні алгоритми кодування (тобто, перетворення послідовностей інформаційних величин на кодові слова) для різних груп значень оцінки ймовірності набуття поточною інформаційною величиною одного з її двох можливих значень. В подальшому переважному варіанті виконання винаходу двійкові кодери 310 втілюють різні алгоритми кодування для різних груп значень для оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення.

В переважному варіанті виконання винаходу двійкові кодери 310 (або один або більша кількість двійкових кодерів) представляють ентропійні кодери, які безпосередньо перетворюють послідовності вхідних інформаційних величин 309 на кодові слова 310. Такі перетворення можуть ефективно виконуватися і не вимагають складного арифметичного кодувального засобу. Обернене перетворення кодових слів на послідовності інформаційних величин (як це робиться в декодері) повинне бути унікальним для гарантії досконалого декодування вхідної послідовності, але перетворення послідовностей 309 інформаційних величин на кодові слова 310 необов'язково потребує бути унікальною, тобто, можна, щоб окрему послідовність інформаційних величин можна було перетворювати на більше ніж одну послідовність кодових слів. В переважному варіанті виконання винаходу перетворення послідовностей вхідних інформаційних величин 309 на кодові слова 310 є бієктивним. В подальшому переважному варіанті виконання винаходу двійкові кодери 310 (або один або більша кількість двійкових кодерів) представляють ентропійні кодери, які безпосередньо перетворюють послідовності вхідних інформаційних величин 309 змінної довжини на кодові слова 310 змінної довжини. В переважному варіанті виконання винаходу вихідні кодові слова представляють коди без надлишку, такі як загальні коди Хаффмана або канонічні коди Хаффмана.

Два приклади для бієктивного перетворення послідовностей інформаційних величин на коди без надлишку зображені в Таблиці 3. В подальшому переважному варіанті виконання винаходу вихідні кодові слова представляють надлишкові коди, придатні для виявлення помилки і виправлення помилки. В подальшому переважному варіанті виконання винаходу вихідні кодові слова представляють шифрувальні коди, придатні для шифрування синтаксичних елементів.

Таблиця 3:

Приклади для перетворень між послідовностями інформаційних величин і кодовими словами

послідовність інформаційних величин (порядок інформаційної величини зліва направо)	кодові слова (порядок бітів зліва направо)
0000 0000	1
0000 0001	0000
0000 001	0001
0000 01	0010
0000 1	0011
0001	0100
001	0101
01	0110
1	0111

послідовність інформаційних величин (порядок інформаційної величини зліва направо)	кодові слова (порядок бітів зліва направо)
000	10
01	11
001	010
11	011
1000 0	0001
1001	0010
1010	0011
1000 1	0000 0
1011	0000 1

- В подальшому переважному варіанті виконання винаходу двійкові кодери 310 (або один або більша кількість двійкових кодерів) представляють ентропійні кодери, які безпосередньо перетворюють послідовності вхідних інформаційних величин 309 змінної довжини на кодові слова 310 фіксованої довжини. В подальшому переважному варіанті виконання винаходу двійкові кодери 310 (або один або більша кількість двійкових кодерів) представляють ентропійні кодери, які безпосередньо перетворюють послідовності вхідних інформаційних величин 309 фіксованої довжини на кодові слова 310 змінної довжини.

- Декодер згідно з варіантом виконання винаходу зображений на Фігурі 8. Декодер виконує головним чином інверсні операції кодера таким чином, що (попередньо кодована) послідовність синтаксичних елементів 327 декодуються з множини з двох або більшої кількості часткових потоків бітів 324. Декодер містить два різні технологічні потоки: потік для інформаційних запитів, який копіює потік даних кодера, і потік даних, який представляє інверсію потоку даних кодера. На Фіг. 8 стрілки, позначені пунктирними лініями, представляють потік інформаційних запитів, тоді як стрілки, позначені суцільною лінією, представляють потік даних. Стандартні блоки декодера головним чином копіюють стандартні блоки кодера, але виконують обернені операції.

- Декодування синтаксичного елемента запускається запитом на надання нового декодованого синтаксичного елемента 313, який надсилається до бінаризатора 314. В переважному варіанті виконання винаходу кожен запит на надання нового декодованого синтаксичного елемента 313 зв'язується з категорією множини однієї або більшої кількості категорій. Категорія, яка зв'язується із запитом на надання синтаксичного елемента, є тією же що й категорія, яка була зв'язана з відповідним синтаксичним елементом під час кодування.

- Бінаризатор 314 перетворює запит на надання синтаксичного елемента 313 на один або більшу кількість запитів на надання інформаційної величини, які надсилаються до присвоювача 316 параметрів. Як кінцеву відповідь на запит на надання інформаційної величини, який надсилається бінаризатором 314 до присвоювача 316 параметрів, бінаризатор 314 приймає декодовану інформаційну величину 326 від двійкового буферного селектора 318. Бінаризатор 314 порівнює прийняту послідовність декодованих двійкових кодів 326 з послідовностями двійкових кодів окремої схеми бінаризації для запрошеного синтаксичного елемента і, якщо прийнята послідовність декодованих двійкових кодів 326 відповідає результатам бінаризації синтаксичного елемента, то бінаризатор спорожнює свій двійковий буфер і видає декодований

синтаксичний елемент як остаточну відповідь по запиту на надання нового декодованого символу. Якщо вже прийнята послідовність декодованих двійкових кодів не відповідає жодній з послідовностей двійкових кодів для схеми бінаризації для запрошеного синтаксичного елемента, то бінаризатор надсилає інший запит на надання інформаційної величини до присвоювання параметрів доти, доки послідовність декодованих двійкових кодів не відповідатиме одній з послідовностей двійкових кодів схеми бінаризації для запрошеного синтаксичного елемента. Для кожного запиту на надання синтаксичного елемента декодер використовує одну і ту ж схему бінаризації, яку використовували для кодування відповідного синтаксичного елемента. Схема бінаризації може відрізнятися для різних категорій синтаксичних елементів. Схема бінаризації для окремої категорії синтаксичного елемента може залежати від множини можливих значень синтаксичного елемента і/або інших властивостей синтаксичних елементів для окремої категорії.

Присвоювач 316 параметрів присвоює множину з одного або більшої кількості параметрів кожному запиту на надання інформаційної величини і надсилає запит на надання інформаційної величини з відповідною множиною параметрів до двійкового буферного селектора. Множина параметрів, які присвоюються запрошеній інформаційній величині присвоювачем параметрів, є тією ж, що була присвоєна відповідній інформаційній величині під час кодування. Множина параметрів може складатися з одного або більшої кількості параметрів, які згадуються в описі кодера з Фіг. 7.

В переважному варіанті виконання винаходу присвоювач 316 параметрів зв'язує кожен запит на надання інформаційної величини з одними і тими ж параметрами що й присвоювач 304 параметрів, тобто, контекст і його відповідне значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень, таке як значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення і ідентифікатор, який специфікує оцінку, завдяки якій одне з двох можливих значень інформаційної величини представляє її менш ймовірне або більш ймовірне значення для поточної запрошеної інформаційної величини.

Присвоювач 316 параметрів може визначати одну або більшу кількість вищезгаданих значень ймовірності (значення оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень, значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення, ідентифікатор, який специфікує оцінку, завдяки якій одне з двох можливих значень інформаційної величини представляє її менш ймовірне або більш ймовірне значення для поточної запрошеної інформаційної величини) на основі множини з одного або більшої кількості вже декодованих символів. Визначення значень ймовірності для окремого запиту на надання інформаційної величини копіює процес в кодері для відповідної інформаційної величини. Декодовані символи, які використовуються для визначення величин ймовірності, можуть включати один або більшу кількість вже декодованих символів однієї і тієї ж категорії, які відповідають множинам даних (таким як блоки або групи зразків) про сусідні просторові і/або тимчасові розташування (по відношенню до множини даних, зв'язаної з поточним запитом на надання синтаксичного елемента), або один або більшу кількість вже декодованих символів різних категорій, які відповідають множинам даних про однакові і/або сусідні просторові, і/або тимчасові розташування (відносно до множини даних, зв'язаної з поточним запитом на надання синтаксичного елемента).

Кожен запит на надання інформаційної величини з відповідною множиною параметрів 317, яка є виходом присвоювача 316 параметрів, подається у двійковий буферний селектор 318. На основі відповідної множини параметрів 317 двійковий буферний селектор 318 надсилає запит на надання інформаційної величини 319 до одного з двох або більшої кількості двійкових буферів 320 і приймає декодовану інформаційну величину 325 від вибраного двійкового буфера 320. Декодована вхідна інформаційна величина 325 потенційно модифікується і декодована вихідна інформаційна величина 326 (з потенційно модифікованим значенням) надсилається до бінаризатора 314 як кінцева відповідь на запит на надання інформаційної величини з відповідною множиною параметрів 317.

Двійковий буфер 320, до якого надсилається запит на надання інформаційної величини, вибирається тим же способом що й двійковий буфер, до якого була надіслана вихідна інформаційна величина двійкового буферного селектора на кодувальній стороні.

В переважному варіанті виконання винаходу двійковий буферний селектор 318 визначає двійковий буфер 320, до якого надсилається запит на надання інформаційної величини 319, на основі відповідного значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень. В переважному варіанті

виконання винаходу множина можливих значень для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень є скінченою, а двійковий буферний селектор 318 містить таблицю, яка точно зв'язує один двійковий буфер 320 з кожним можливим значенням оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень, де різні значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень можуть зв'язуватися з одним і тим же двійковим буфером 320. В подальшому переважному варіанті виконання винаходу інтервал можливих значень для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень ділиться на ряд інтервалів, двійковий буферний селектор 318 визначає індекс інтервалу для поточної оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень і двійковий буферний селектор 318 містить таблицю, яка точно зв'язує один двійковий буфер 320 з кожним можливим значенням для індексу інтервалу, де різні значення для індексу інтервалу можуть зв'язуватися з одним і тим же двійковим буфером 320. В переважному варіанті виконання винаходу запити на надання інформаційних величин 317 з протилежними значеннями для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень (протилежні значення є тими, які представляють оцінки ймовірностей P і $1 - P$) надсилаються до одного і того ж двійкового буфера 320. В подальшому переважному варіанті виконання винаходу зв'язування значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень з конкретним двійковим буфером адаптується з часом.

В подальшому переважному варіанті виконання винаходу двійковий буферний селектор 318 визначає двійковий буфер 320, до якого надсилається запит на надання інформаційної величини 319, на основі відповідного значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення. В переважному варіанті виконання винаходу множина можливих значень для оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення є скінченою і двійковий буферний селектор 318 містить таблицю, яка точно зв'язує один двійковий буфер 320 з кожним можливим значенням оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення, де різні значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення можуть зв'язуватися з одним і тим же двійковим буфером 320. В подальшому переважному варіанті виконання винаходу інтервал можливих величин для оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення ділиться на ряд інтервалів, двійковий буферний селектор 318 визначає індекс інтервалу для поточної оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення і двійковий буферний селектор 318 містить таблицю, яка точно зв'язує один двійковий буфер 320 з кожним можливим значенням для індексу інтервалу, де різні значення для індексу інтервалу можуть зв'язуватися з одним і тим же двійковим буфером 320. В подальшому переважному варіанті виконання винаходу зв'язування оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення з окремим двійковим буфером адаптується з часом.

Після одержання декодованої інформаційної величини 325 від вибраного двійкового буфера 320 двійковий буферний селектор 318 потенційно модифікує вхідну інформаційну величину 325 і надсилає вихідну інформаційну величину 326 (з потенційно модифікованим значенням) до бінаризатора 314. Перетворення вхідної/вихідної інформаційної величини двійкового буферного селектора 318 є оберненим перетворенням вхідної/вихідної інформаційної величини двійкового буферного селектора на кодувальній стороні.

В переважному варіанті виконання винаходу двійковий буферний селектор 318 не змінює значення інформаційної величини, тобто, вихідна інформаційна величина 326 завжди має одне і те ж значення що й вхідна інформаційна величина 325. В подальшому переважному варіанті виконання винаходу двійковий буферний селектор 318 визначає значення вихідної інформаційної величини 326 на основі значення вхідної інформаційної величини 325 і значення для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень, яке зв'язане із запитом на надання інформаційної величини 317. В переважному варіанті виконання винаходу значення вихідної інформаційної величини 326 встановлюється рівним значенню вхідної інформаційної величини 325, якщо значення ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень менше ніж (або менше ніж або дорівнює) окреме порогове значення; якщо

значення ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень більше або дорівнює (або більше за) окремому пороговому значенню, то значення вихідної інформаційної величини 326 модифікується (тобто, воно встановлюється рівним протилежному значенню вхідної інформаційної величини). В подальшому переважному варіанті виконання винаходу значення вихідної інформаційної величини 326 встановлюється рівним значенню вхідної інформаційної величини 325, якщо значення ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень більше за (або більше за або дорівнює) окреме порогове значення; якщо значення ймовірності на набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень менше ніж або дорівнює (або менше ніж) окремому пороговому значенню, то значення вихідної інформаційної величини 326 модифікується (тобто, воно встановлюється рівним протилежному значенню вхідної інформаційної величини). В переважному варіанті виконання винаходу значення порогу відповідає величині 0,5 для оціненої ймовірності набуття поточною запрошеною інформаційною величиною обох її можливих величин.

В подальшому переважному варіанті виконання винаходу двійковий буферний селектор 318 визначає значення вихідної інформаційної величини 326 на основі значення вхідної інформаційної величини 325, а ідентифікатор, який специфікує оцінку, завдяки якій одне з двох можливих значень інформаційної величини представляє її менш ймовірне або більш ймовірне значення для поточної запрошеної інформаційної величини, яке зв'язане із запитом на надання інформаційної величини 317. В переважному варіанті виконання винаходу значення вихідної інформаційної величини 326 встановлюється рівним значенню вхідної інформаційної величини 325, якщо ідентифікатор специфікує, що перше з двох можливих значень інформаційної величини представляє її менш ймовірне (або більш ймовірне) значення для поточної запрошеної інформаційної величини і значення вихідної інформаційної величини 326 модифікується (тобто, воно встановлюється рівним протилежному значенню вхідної інформаційної величини), якщо ідентифікатор специфікує, що друге з двох можливих значень інформаційної величини представляє її менш ймовірне (або більш ймовірне) значення для поточної запрошеної інформаційної величини.

Як описано вище, двійковий буферний селектор надсилає запит на надання інформаційної величини 319 одному з двох або більшої кількості двійкових буферів 320. Двійкові буфери 20 являють собою буфери, які працюють по принципу „першим прийшов першим вийшов“, до яких подаються послідовності декодованих інформаційних величин 321 від під'єднаних двійкових декодерів 322. Як відповідь на запит на надання інформаційної величини 319, яка надсилається до двійкового буфера 320 від двійкового буферного селектора 318, двійковий буфер 320 видаляє інформаційну величину з свого контенту, який спершу був поданий в двійковий буфер 320, і надсилає його до двійкового буферного селектора 318. Інформаційні величини, які раніше надсилаються до двійкового буфера 320, раніше видаляються і надсилаються до двійкового буферного селектора 318.

Кожен з двох або більшої кількості двійкових буферів 320 точно з'єднаний з одним двійковим декодером 322 і кожен двійковий декодер з'єднаний тільки з одним двійковим буфером 320. Кожен двійковий декодер 322 зчитує кодові слова 323, які представляють послідовності бітів, з окремого часткового потоку бітів 324. Двійковий декодер перетворює кодове слово 323 на послідовність інформаційних величин 321, яка надсилається до під'єданого двійкового буфера 320. Загальний алгоритм декодування перетворює два або більшу кількість часткових потоків бітів 324 на ряд декодованих синтаксичних елементів, де кількість часткових потоків бітів дорівнює кількості двійкових буферів і двійкових декодерів, а декодування синтаксичних елементів запускається запитом на надання нових синтаксичних елементів. В переважному варіанті виконання винаходу двійковий декодер 322 перетворює кодові слова 323 із змінною кількістю бітів на послідовність змінної кількості інформаційних величин 321. Одна перевага варіантів виконання винаходу полягає в тому, що декодування інформаційних величин з двох або більшої кількості часткових потоків бітів може виконуватися паралельно (наприклад, для різних груп значень ймовірності), що зменшує час обробки для декількох втілень.

Інша перевага варіантів виконання винаходу полягає в тому, що декодування інформаційних величин, яке виконується двійковими декодерами 322, може спеціально розроблятися для різних множин параметрів 317. Зокрема, кодування і декодування інформаційних величин можуть оптимізуватися (з точки зору ефективності кодування і/або складності) для різних груп оцінених ймовірностей. З одного боку, це дозволяє послаблення складності кодування/декодування відносно алгоритмів ентропійного кодування рівня техніки з подібною ефективністю кодування. З іншого боку, це дозволяє покращити ефективність кодування відносно алгоритмів ентропійного кодування рівня техніки з подібною складністю

кодування/декодування. В переважному варіанті виконання винаходу двійкові декодери 322 втілюють різні алгоритми декодування (тобто, перетворення послідовностей інформаційних величин на кодові слова) для різних груп значень для оцінки ймовірності набуття поточною запрошеною інформаційною величиною одного з двох її можливих значень 317. В подальшому

5 переважному варіанті виконання винаходу двійкові декодери 322 втілюють різні алгоритми декодування для різних груп значень для оцінки ймовірності набуття поточною запрошеною інформаційною величиною її менш ймовірного або більш ймовірного значення.

Двійкові декодери 322 виконують обернене перетворення відносно до перетворення, здійснюваного відповідними двійковими кодерами на кодувальній стороні.

10 В переважному варіанті виконання винаходу двійкові декодери 322 (або один або більша кількість двійкових декодерів) представляють ентропійні декодери, які безпосередньо перетворюють кодові слова 323 на послідовності інформаційних величин 321. Такі перетворення можуть ефективно виконуватися і не вимагають складного арифметичного кодувального засобу. Перетворення кодових слів на послідовності інформаційних величин

15 повинно бути унікальним. В переважному варіанті виконання винаходу перетворення кодових слів 323 на послідовності інформаційних величин 321 є бієктивним. В подальшому переважному варіанті виконання винаходу двійкові декодери 310 (або один або більша кількість двійкових декодерів) представляють ентропійні декодери, які безпосередньо перетворюють кодові слова 323 змінної довжини на послідовності інформаційних величин 321 змінної довжини. В

20 переважному варіанті виконання винаходу вхідні кодові слова представляють коди без надлишку, такі як загальні коди Хаффмана або канонічні коди Хаффмана. Два приклади для бієктивного перетворення кодів без надлишку на послідовності інформаційних величин вказані в Таблиці 3.

В подальшому переважному варіанті виконання винаходу двійкові декодери 322 (або один

25 або більша кількість двійкових декодерів) представляють ентропійні декодери, які безпосередньо перетворюють кодові слова 323 фіксованої довжини на послідовності інформаційних величин 321 змінної довжини. В подальшому переважному варіанті виконання винаходу двійкові декодери 322 (або один або більша кількість двійкових декодерів) представляють ентропійні декодери, які безпосередньо перетворюють кодові слова 323 змінної

30 довжини на послідовності інформаційних величин 321 фіксованої довжини.

Таким чином, Фіг. 7 і 8 зображають варіант виконання кодера для кодування послідовності символів 3 і декодер для їх відновлення. Кодер містить присвоювач 304, сконфігурований для присвоєння ряду параметрів 305 кожному символу послідовності символів. Присвоєння базується на інформації, яка міститься в попередніх символах послідовності символів, такий як

35 категорія синтаксичного елемента 1 для представлення, такого як бінаризація, якому належить поточний символ і яке, згідно з синтаксичною структурою синтаксичних елементів 1, на даний момент очікується, очікування якого, у свою чергу, можна одержати з історії попередніх синтаксичних елементів 1 і символів 3. Окрім того, кодер містить множину ентропійних кодерів 10, кожен з яких сконфігурований для перетворення символів 3, надісланих до відповідного

40 ентропійного кодера, на відповідний потік бітів 312, і селектор 306, сконфігурований для надсилання кожного символу 3 до вибраного одного з множини ентропійних кодерів 10, при цьому вибір залежить від кількості параметрів 305, присвоєних відповідному символу 3. Присвоювач 304 міг би передбачатися інтегрованим в селектор 206 для одержання відповідного селектора 502.

45 Декодер для відновлення послідовності символів містить множину ентропійних декодерів 322, кожен з яких сконфігурований для перетворення відповідного потоку бітів 323 на символи 321; присвоювач 316, сконфігурований для присвоєння ряду параметрів 317 кожному символу 315 послідовності символів, які відновлюються на основі інформації, яка міститься в попередньо відновлених символах послідовності символів (дивіться позиції 326 і 327 на Фіг. 8); і селектор

50 318, сконфігурований для одержання кожного символу послідовності символів, які відновлюються, з вибраного одного з множини ентропійних кодерів 322, при цьому вибір залежить від кількості параметрів, визначених для відповідного символу. Присвоювач 316 може конфігуруватися так, що кількість параметрів, присвоєних кожному символу, містить або є засобом для оцінки ймовірності розподілу можливих значень символу, які він може припускати.

55 Знову, присвоювач 316 і селектор 318 можуть передбачатися як інтегровані в один блок - селектор 402. Послідовність символів, які відновлюються, може бути бінарним алфавітом і присвоювач 316 може конфігуруватися так, що оцінка розподілу ймовірності складається із значення для оцінки ймовірності набуття поточною інформаційною величиною її менш ймовірного або більш ймовірного значення з двох її можливих значень бінарного алфавіту, а

60 ідентифікатор, який специфікує оцінку, завдяки якій одне з двох можливих значень

інформаційної величини представляє її менш ймовірне або більш ймовірне значення. Присвоювач 316 може додатково конфігуруватися для внутрішнього присвоювання контексту кожному символу послідовності символів 315, які відновлюються на основі інформації, яка міститься в попередньо відновлених символах послідовності символів, які відновлюються, при цьому кожен контекст має відповідну зв'язану з ним оцінку розподілу ймовірності, і для адаптації оцінки розподілу ймовірності для кожного контексту до реальної символної статистики на основі значень попередньо відновлених символів, якій присвоєний відповідний контекст. Контекст може брати до уваги просторовий зв'язок або сусідство положень, яким належать синтаксичні елементи, такі як у кодуванні відеоданих або кодуванні картинки, або навіть в таблицях у випадку застосувань у фінансовій сфері. Потім, значення для оцінки розподілу ймовірності для кожного символа може визначатися на основі оцінки розподілу ймовірності, зв'язаної з контекстом, присвоєного відповідному символу, як наприклад за допомогою дискретизації, або використовуючи індекс у відповідній таблиці, на основі оцінки розподілу ймовірності, зв'язаної з контекстом, присвоєного відповідним символом (у нижченаведених варіантах виконання, індексованих ріре індексом разом з індексом деталізації) одному з множини представників оцінки розподілу ймовірності (відрізняється від індексу деталізації) для одержання значення оцінки розподілу ймовірності (ріре індекс, який позначає частковий потік бітів 312). Селектор може конфігуруватися так, що між множиною ентропійних кодерів і множиною представників оцінки розподілу ймовірності визначається бієктивний зв'язок. Селектор 18 може конфігуруватися для зміни дискретного перетворення з інтервалу оцінок розподілу ймовірності на множину представників оцінки розподілу ймовірності наперед встановленим детерміністичним способом в залежності від попередньо відновлених символів послідовності символів протягом часу. Тобто, селектор 318 може змінювати розміри кроку дискретизації, тобто, інтервали розподілу ймовірності, перетворені на окремі індекси ймовірності, бієктивно зв'язані з окремими ентропійними декодерами. Множина ентропійних декодерів 322, у свою чергу, може конфігуруватися для адаптації свого способу перетворення символів на потоки бітів, яка відповідає за зміну в дискретному перетворенні. Наприклад, кожен ентропійний декодер 322 може оптимізуватися, тобто, може мати оптимальний коефіцієнт стиснення для певної оцінки розподілу ймовірності у відповідному інтервалі дискретизації оцінки розподілу ймовірності і може змінювати перетворення своєї послідовності кодових слів/символів для адаптації положення цієї певної оцінки розподілу ймовірності у відповідному інтервалі дискретизації оцінки розподілу ймовірності при зміні останнього для його оптимізації. Селектор може конфігуруватися для зміни дискретного перетворення так, що швидкості, з якими символи виділяються з множини ентропійних декодерів, робляться менш розкиданими. Щодо бінаризатора 314, відмічається, що він може усуватися, якщо синтаксичні елементи є вже двійковими. Окрім того, в залежності від типу декодера 322, існування буферів 320 не потрібне. Окрім того, буфери можуть інтегруватися в декодери.

Закінчення скінчених послідовностей синтаксичних елементів

В переважному варіанті виконання винаходу кодування і декодування виконується для скінченної множини синтаксичних елементів. Часто кодується певна кількість даних, таких як нерухоме зображення, кадр або поле послідовності відеоданих, частина зображення, частина кадру або поля послідовності відеоданих, або множина послідовних аудіозразків і так далі. Для скінчених множин синтаксичних елементів, головним чином, часткові потоки бітів, які створюються на кодувальній стороні, повинні закінчуватися, тобто, слід гарантувати можливість декодування усіх синтаксичних елементів з переданих або збережених часткових потоків бітів. Після введення останньої інформаційної величини у відповідний двійковий буфер 308, двійковий кодер 310 повинен гарантувати запис повного кодового слова в частковий потік бітів 312. Якщо двійковий кодер 310 представляє ентропійний кодер, який втілює пряме перетворення послідовностей інформаційних величин на кодові слова, то послідовність інформаційних величин, яка зберігається в двійковому буфері після запису останньої інформаційної величини у двійковий буфер, може не представляти послідовність інформаційних величин, яка зв'язується з кодовим словом (тобто, вона могла б представляти префікс двох або більшої кількості послідовностей інформаційних величин, які зв'язуються з кодовими словами). У такому випадку, будь-яке з кодових слів, зв'язане з послідовністю інформаційних величин, яке містить послідовність інформаційних величин у двійковому буфері як префікс, повинно записуватися в частковий потік бітів (двійковий буфер повинен очищатися). Це могло б виконуватися вставлянням інформаційних величин з окремим або довільним значенням у двійковий буфер, доки не буде записане кодове слово. В переважному варіанті виконання винаходу двійковий кодер вибирає одне з кодових слів з мінімальною довжиною (на додаток до властивості, яка полягає в тому, що відповідна послідовність інформаційних величин повинна

містити послідовність інформаційних величин у двійковому буфері як префікс). На декодувальній стороні двійковий декодер 322 може декодувати більше інформаційних величин ніж вимагається для останнього кодового слова в частковому потоці бітів; ці інформаційні величини не запрошуються двійковим буферним селектором 318 і викидаються та ігноруються.

5 Декодування скінченної множини символів контролюється запитами на надання декодованих синтаксичних елементів; якщо не запрошується додатковий синтаксичний елемент для певної кількості даних, то декодування припиняється.

Передача і мультимплексування часткових потоків бітів

10 Часткові потоки бітів 312, які створюються кодером, можуть передаватися окремо або вони можуть мультимплексуватися в єдиний потік бітів, або кодові слова часткових потоків бітів можуть чергуватися в єдиному потоці бітів.

У варіанті виконання винаходу кожен частковий потік бітів для певної кількості даних записується в один пакет даних. Кількість даних може бути довільною множиною синтаксичних елементів, таких як нерухоме зображення, поле або кадр послідовності відеоданих, частина 15 нерухомого зображення, частина поля або кадру послідовності відеоданих, або основа аудіовибіроч і так далі.

В іншому переважному варіанті виконання винаходу два або більша кількість часткових потоків бітів для певної кількості даних або усі часткові потоки бітів для певної кількості даних мультимплексуються в один пакет даних. Структура пакета даних, який містить мультимплексовані 20 часткові потоки бітів, зображений на Фігурі 9.

Пакет даних 400 складається із заголовка і одного розділу для даних кожного часткового потоку бітів (для розглядуваної кількості даних). Заголовок 400 пакета даних містить індикатори для поділу (решти) пакета даних на сегменти даних 402 потоку бітів. Окрім індикаторів для поділу заголовка може містити додаткову інформацію. У переважному варіанті виконання 25 винаходу індикатори для поділу пакета даних є місцями початку сегментів даних, виражених у бітах або байтах, або множин бітів, або множин байтів. В переважному варіанті виконання винаходу розташування початку сегментів даних кодуються як абсолютні величини в заголовці пакета даних або відносно початку пакета даних або відносно кінця заголовка, або відносно початку попереднього пакета даних. В подальшому переважному варіанті виконання винаходу 30 розташування початку сегментів даних кодуються диференціально, тобто, кодується тільки різниця між реальним початком сегмента даних і прогнозом для початку сегменту даних. Прогноз може одержуватися на основі вже відомої або переданої інформації, такої як загальний розмір пакета даних, розмір заголовку, кількість сегментів даних в пакеті даних, розташування початку попередніх сегментів даних. В переважному варіанті виконання винаходу розташування 35 початку першого пакета даних не кодується, а кодується передбачуване розташування на основі розміру заголовка пакету даних. На декодувальній стороні передані індикатори поділу використовуються для одержання інформації про початок сегментів даних. Сегменти даних потім використовуються як часткові потоки бітів і дані, які містяться в сегментах даних, надсилаються у відповідні двійкові декодери у послідовному порядку.

40 Існує декілька альтернатив для мультимплексування часткових потоків бітів з одержанням пакета даних. Одна альтернатива, яка може послабити потребу у побічній інформації, зокрема для випадків, у яких розміри часткових потоків бітів дуже подібні, зображена на Фіг. 10. Корисне навантаження пакета даних, тобто, пакет даних 410 без свого заголовка 411, ділиться на сегменти 412 наперед визначеним способом. Як приклад, корисне навантаження пакета даних 45 може ділитися на сегменти однакового розміру. Потім кожен сегмент зв'язується з частковим потоком бітів або з першою частиною часткового потоку бітів 413. Якщо частковий потік бітів більший за відповідний сегмент даних, то його залишок 414 поміщається у невикористаний простір в кінці інших сегментів даних. Це може виконуватися у такий спосіб, що решта потоку бітів вставляється в зворотному порядку (починаючи з кінця сегмента даних), що зменшує 50 побічну інформацію. Зв'язування решт часткових потоків бітів з сегментами даних і, коли до сегмента даних додається більше ніж один залишок, то початкова точка для одного або більшої кількості залишків повинна сигналізуватися всередині потоку бітів, наприклад в заголовку пакета даних.

Чергування кодових слів змінної довжини

55 Для деяких застосувань вищеописане мультимплексування часткових потоків бітів (для певної кількості синтаксичних елементів) в одному пакеті даних може мати наступні недоліки: з одного боку, для малих пакетів даних кількість бітів для побічної інформації, яка вимагається для сигналізації поділу, може стати значущою по відношенню до реальних даних в часткових потоках бітів, що, врешті решт, знижує ефективність кодування. З іншого боку, 60 мультимплексування може не підходити для застосувань, які вимагають малої затримки

(наприклад, проведення відеоконференцій). За допомогою описаного мультиплексування кодер не може починати передачу пакета даних перед повним формуванням часткових потоків бітів, оскільки розташування початку поділів перед цим не відомі. Окрім того, головним чином, декодер повинен чекати до тих пір, доки він не прийме початок останнього сегмента даних перед тим, як він зможе розпочати декодування пакета даних. Для застосувань у формі систем для проведення відеоконференцій, ці затримки можуть робити внесок у додаткову загальну затримку системи з декількох відеокартинок (зокрема, для швидкостей передачі бітів, які близькі до швидкості передачі даних, і для кодерів/декодерів, які вимагають часового інтервалу між двома картинками для кодування/декодування картинки), що є важливим для таких застосувань. Для усунення недоліків певних застосувань, кодер переважного варіанта виконання винаходу може конфігуруватися у такий спосіб, що кодові слова, які генеруються двома або більшою кількістю двійкових кодерів, чергуються в єдиному потоці бітів. Потік бітів з чергованими кодовими словами може безпосередньо надсилатися до декодера (при нехтуванні малою затримкою буфера, дивіться нижче). На декодувальній стороні два або більша кількість двійкових декодерів зчитують кодові слова безпосередньо з потоку бітів в порядку декодування; декодування може починатися з першим прийнятим бітом. Окрім того, не вимагається побічної інформації для сигналізації мультиплексування (або чергування) часткових потоків бітів. Подальше послаблення складності декодера може досягатися, коли двійкові декодери 322 не зчитують кодові слова змінної довжини з глобального бітового буфера, а замість цього вони завжди зчитують послідовності бітів фіксованої довжини з глобального бітового буфера і додають ці послідовності бітів фіксованої довжини до локального бітового буфера, де кожен двійковий декодер 322 з'єднаний з окремим локальним бітовим буфером. Кодові слова змінної довжини потім зчитуються з локального бітового буфера. Тому, синтаксичний аналіз кодових слів змінної довжини може виконуватися паралельно, тільки доступ до послідовностей бітів фіксованої довжини повинен виконуватися синхронно, але такий доступ до послідовностей бітів фіксованої довжини зазвичай є дуже швидким таким чином, що загальна складність декодування може послаблюватися для деяких архітектур. Фіксована кількість інформаційних величин, які надсилаються до окремого локального бітового буфера, може бути іншою для іншого локального бітового буфера і вона може також змінюватися з часом в залежності від певних параметрів як подій в двійковому декодері, двійковому буфері або бітовому буфері. Однак, кількість бітів, які зчитуються окремим доступом, не залежать від реальних бітів, які зчитуються під час окремого доступу, що є важливою відмінністю для зчитування кодових слів змінної довжини.

Зчитування послідовностей бітів фіксованої довжини запускається певними подіями в двійкових буферах, двійкових декодерах або локальних бітових буферах. Як приклад, можна запрошувати зчитування нової послідовності бітів фіксованої довжини, коли кількість бітів, які присутні у підключеному бітовому буфері, падає нижче наперед встановленої порогової величини, де різні порогові величини можуть використовуватися для різних бітових буферів. В кодері потрібно гарантувати, щоб послідовності інформаційних величин фіксованої довжини вставлялися в одному і тому ж порядку в потік бітів, у якому вони зчитуються з потоку бітів на декодувальній стороні. Також можна поєднувати це чергування послідовностей бітів фіксованої довжини з контролем малої затримки, подібного до пояснених вище. Далі, описується переважний варіант виконання для чергування послідовностей бітів фіксованої довжини. За подальшими деталями, які стосуються останніх схем чергування, звертайтеся до документа WO2011/128268A1.

Після опису варіантів виконання, згідно з якими навіть попереднє кодування використовується для стискання відеоданих, описується навіть подальший варіант виконання представленого винаходу, який робить втілення особливо ефективним з точки зору гарного компромісу між коефіцієнтом стискання, з одного боку, і таблицею пошуку та додатковими обрахунками, з іншого боку. Зокрема, наступні варіанти виконання дозволяють використання з обрахункової точки зору менш складних кодів змінної довжини для індивідуального ентропійного кодування потоків бітів і ефективного охоплення частин оцінки ймовірності. В описаних нижче варіантах виконання символи є бінарними і коди змінної довжини (VLC), представлені нижче, ефективно охоплюють оцінку ймовірності, представлену, наприклад R_{LPS} , яке розташоване в інтервалі $[0;0,5]$.

Зокрема, вказані нижче варіанти виконання описують можливі втілення для індивідуальних ентропійних кодерів 310 і декодерів 322, відповідно, на Фіг. 7-17. Вони придатні для кодування інформаційних величин, тобто бінарних символів, оскільки вони трапляються в стисканні зображення або відеоданих. Відповідно, ці варіанти виконання також застосовуються для кодування зображення або відеоданих, де такі бінарні символи діляться на один або більшу

кількість потоків інформаційних величин 307, які кодуються, і, відповідно, потоки бітів 324, які декодуються, де кожен такий потік інформаційних величин може розглядатися як реалізація процесу Бернуллі. Описані нижче варіанти виконання використовують один або більшу кількість пояснених нижче різних способів так званого кодування з перетворенням послідовності символів змінної довжини на послідовність бітів змінної довжини (v2v-коди) для кодування потоків інформаційних величин. v2v-код може розглядатися як два коди без префіксу з однією і тією ж кількістю кодових слів: головний і допоміжний код без префіксу. Кожне кодове слово головного коду без префіксу зв'язується з одним кодовим словом допоміжного коду без префіксу. У відповідності з нижчеописаними варіантами виконання принаймні деякі з кодерів 310 і декодерів 322 працюють наступним чином: Для кодування окремої послідовності інформаційних величин 307 кожен раз, коли кодове слово головного коду без префіксу зчитується з буфера 308, відповідне кодове слово допоміжного коду без префікса записується в потік бітів 312. Та ж сама процедура використовується для декодування такого потоку бітів 24, але з чергуванням головного і допоміжного коду без префікса. Тобто, для декодування потоку бітів 324, кожен раз, коли кодове слово допоміжного коду без префіксу зчитується з відповідного потоку бітів 324, відповідне кодове число допоміжного коду без префіксу записується в буфер 320.

Переважно, описані нижче коди не потребують таблиць пошуку. Коди здатні втілюватися у формі машин з кінцевим числом станів. Представлені тут V2V-коди можуть генеруватися простими правилами конструювання так, що для кодових слів не потрібно зберігати великі таблиці. Замість цього, може використовуватися простий алгоритм для виконання кодування або декодування. Нижче описуються три правила формування, де два з них можуть параметризуватися. Вони охоплюють різні або навіть окремі частини вищезгаданого інтервалу ймовірності і є, відповідно, особливо вигідними, коли використовуються разом, так як усі три коди або два з них використовуються паралельно (кожен для різних кодерів/декодерів 11 і 22). За допомогою описаних нижче правил формування можна формувати множину V2V-кодів так, що для процесів Бернуллі з довільною ймовірністю p один з кодів гарно працює з точки зору надмірної довжини коду.

Як визначено вище, кодування і декодування потоків 312 і, відповідно, 324 може виконуватися або незалежно для кожного потоку або поперемінно. Однак, це не є спеціальним для представлених класів v2v-кодів і, тому, далі описується тільки кодування і декодування окремого кодового слова для кожного з трьох правил формування. Однак, підкреслюється, що усі вищезгадані варіанти виконання, які стосуються рішень з поперемінним виконанням, також здатні поєднуватися з на даний момент описаними кодами або кодерами і декодерами 310 і, відповідно, 322.

Правило формування 1: «унарні двійкові ріре» коди або кодери/декодери 310 і 322 Унарні двійкові ріре коди (PIPE = ентропія поділу інтервалу ймовірності) є спеціальним варіантом так званих «двійкових ріре» кодів, тобто, кодів, придатних для кодування будь-якого з окремих потоків бітів 12 і 24, кожен з яких переносить дані статистики бінарних символів, яка належить певному підінтервалу ймовірності вищезгаданого інтервалу ймовірності $[0,0,5]$. Спершу описується формування двійкових ріре кодів. Двійковий ріре код може формуватися з будь-якого коду без префіксу з принаймні трьома кодовими словами. Для формування v2v-коду, використовують код без префіксу як головний і допоміжний код, але з чергуванням двох кодових слів допоміжного коду без префіксу. Це означає, що, за виключенням двох кодових слів, інформаційні величини записуються в незмінний потік бітів. За допомогою цієї технології необхідно зберігати тільки один код без префіксу разом з інформацією, які два кодові слова чергуються між собою, і, таким чином, використання пам'яті зменшується. Відзначається, що доцільно обмінюватися тільки кодовими словами різної довжини, оскільки, інакше, потік бітів повинен мати ту ж довжину що й потік інформаційних величин (нехтуючи ефектами, які трапляються в кінці потоку інформаційних величин).

Завдяки цьому правилу формування чудовою властивістю двійкових ріре кодів є те, що, коли головний і допоміжний код без префіксу міняються між собою (тоді як перетворення кодових слів зберігається), то одержуваний v2v-код ідентичний оригінальному v2v-коду. Тому, алгоритм кодування і алгоритм декодування ідентичні для двійкових ріре кодів.

Унарний двійковий ріре код формується з спеціального коду без префіксу. Цей спеціальний код без префіксу формується наступним чином. Спершу, генерують код без префіксу, який складається з p унарних кодових слів, починаючи з '01', '001', '0001', ... доти, доки не одержиться p кодових слів, p є параметром для унарного двійкового ріре коду. З найдовшого кодового слова видаляється слід 1. Це відповідає зрізаному унарному коду (але без кодового слова '0'). Потім, генеруються $p-1$ унарних кодових слів, починаючи з '10', '110', '1110', ... доти, доки не буде

створено $n-1$ кодових слів. З найдовшого з цих кодових слів видаляють слід 0. Об'єднана множина цих двох кодів без префікса використовується як вхідні дані для генерування унарного двійкового ріре коду. Два кодові слова, які міняються між собою, є тими, де одне з них складається тільки з 0, а інше складається тільки з 1.

5

Приклад для $n = 4$:

№	Головний	Допоміжний
1	0000	111
2	0001	0001
3	001	001
4	01	01
5	10	10
6	110	110
7	111	0000

Правило формування 2: коди, які відображають перетворення «унарних кодів на коди Райса» і кодери/декодери 10 і 22, які використовують перетворення унарних кодів на коди Райса:

10

Коди, які відображають перетворення унарних кодів на коди Райса, використовують зрізаний унарний код як головний код. Тобто, унарні кодові слова генеруються починаючи з '1', '01', '001', ... доти, доки не буде згенеровано $2^n + 1$ кодових слів і з найдовшого кодового слова видаляють слід 1. n є параметром коду, який відображає перетворення унарного коду на код Райса. Допоміжний код без префіксу формується з кодових слів головного коду без префіксу наступним чином. Головному кодовому слову, яке складається тільки з 0, присвоюється кодове слово '1'. Усі інші кодові слова складаються з конкатенації кодового слова '0' з n -бітовим двійковим представленням ряду 0 відповідного кодового слова головного коду без префіксу.

15

20

Приклад для $n=3$:

№	Головний	Допоміжний
1	1	0000
2	01	0001
3	001	0010
4	0001	0011
5	00001	0100
6	000001	0101
7	0000001	0110
8	00000001	0111
9	00000000	1

Відзначаємо, що це ідентичне перетворенню нескінченного унарного коду на код Райса з параметром Райса $2n$.

Правило формування 3:

25

'Потрійний двійковий' код

Потрійний двійковий код надається наступним чином:

№	Головний	Допоміжний
1	000	0
2	001	100
3	010	101
4	100	110
5	110	11100
6	101	11101
7	011	11110
8	111	11111

Він має ту властивість, що головний код (послідовності символів) має фіксовану довжину (завжди три інформаційні величини), а кодові слова відбираються по висхідних кількостях 1.

30

Далі описується ефективно втілення потрійного двійкового коду. Кодер і декодер для потрійного двійкового коду можуть втілюватися без зберігання таблиць наступним чином.

В кодері (будь-який з кодерів 10) три інформаційні величини зчитуються з потоку інформаційних величин (тобто, з потоку 7). Якщо ці три інформаційні величини містять точно одну 1, то кодове слово '1' записується в потік бітів після двох інформаційних величин, які складаються з двійкового представлення положення 1 (починаючи з права з 00). Якщо три

5 інформаційні величини містять точно один 0, то кодове слово '111' записується в потік бітів після двох інформаційних величин, які складаються з двійкового представлення положення 0 (починаючи справа з 00). Решта кодових слів '000' і '111' перетворюються на '0' і, відповідно, '11111'.

В декодері (будь-який з декодерів 22) одна інформаційна величина або біт зчитується з відповідного потоку бітів 24. Якщо вона дорівнює '0', то кодове слово '000' декодується з одержанням потоку інформаційних величин 21. Якщо вона дорівнює '1', то дві або більша кількість інформаційних величин зчитуються з потоку бітів 24. Якщо ці два біти не дорівнюють '11', то вони інтерпретуються як двійкове представлення числа і два 0 та одна 1 декодується з одержанням потоку бітів так, що положення 1 визначається числом. Якщо два біти дорівнюють

15 '11', то два додаткових біти зчитуються і інтерпретуються як двійкове представлення числа. Якщо це число менше ніж 3, то дві 1 і один 0 декодується і число визначає положення 0. Якщо воно дорівнює 3, то '111' декодується з одержанням потоку інформаційних величин.

Далі описується ефективне втілення унарних двійкових ріре кодів. Кодер і декодер для унарних двійкових ріре кодів можуть ефективно втілюватися шляхом використання лічильника. Завдяки структурі двійкових ріре кодів кодування і декодування двійкових ріре кодів легко втілювати:

20

В кодері (будь-який з кодерів 10), якщо перша інформаційна величина кодового слова дорівнює '0', то інформаційні величини обробляються до появи '1' або до зчитування n 0 (включаючи перший '0' кодового слова). Якщо трапилася '1', то зчитані інформаційні величини записуються в незмінний потік бітів. Інакше (тобто, зчитано n 0), то $n-1$ 1 записується в потік бітів. Якщо перша інформаційна величина кодового слова дорівнює '1', то інформаційні величини обробляються доки не трапиться '0' або доки не зчитується $n-1$ 1 (включаючи першу '1' кодового слова). Якщо трапився '0', то зчитані інформаційні величини записуються в незмінний потік бітів. Інакше (тобто, зчитано $n-1$ 1), n 0 записуються в потік бітів.

25

В декодері (будь-який з декодерів 322) використовується той же алгоритм для кодера, оскільки це застосовується для двійкових ріре кодів, як описано вище.

30

Далі описується ефективне втілення кодів, які відображають перетворення унарних кодів на коди Райса. Кодер і декодер для кодів, які відображають перетворення унарних кодів на коди Райса, можуть ефективно втілюватися шляхом використання лічильника, як тепер буде описуватися.

35

В кодері (будь-який з кодерів 310) інформаційні величини зчитуються з потоку інформаційних величин (тобто, потоку 7) доки не трапиться 1 або доки не буде зчитано $2n$ 0. Кількість 0 підраховується. Якщо підрахована кількість дорівнює $2n$, то кодове слово '1' записується в потік бітів. Інакше, '0' записується перед двійковим представленням підрахованої кількості записаної з n бітами.

40

В декодері (будь-який з декодерів 322) зчитується один біт. Якщо він дорівнює '1', то $2n$ 0 декодується з одержанням рядка інформаційних величин. Якщо він дорівнює '0', то зчитується n додаткових бітів, які інтерпретуються як двійкове представлення числа. Ця кількість 0 декодується з одержанням потоку інформаційних величин, за яким слідує '1'.

45

Іншими словами тільки що описані варіанти виконання розкривають кодер для кодування послідовності символів 303, який містить присвоювач 316, сконфігурований для присвоювання ряду параметрів 305 кожному символу послідовності символів на основі інформації, яка міститься в попередніх символах послідовності символів; певну кількість ентропійних кодерів 310, кожен з яких сконфігурований для перетворення символів 307, надісланих до відповідного ентропійного кодера 310, на відповідний потік бітів 312; і селектор 6, сконфігурований для надсилання кожного символа 303 до вибраного одного з множини ентропійних кодерів 10, при цьому вибір залежить від кількості параметрів 305, присвоєних відповідному символу 303. Згідно з тільки що зазначеними варіантами виконання принаймні перша підмножина ентропійних кодерів може бути кодером змінної довжини, сконфігурованим для перетворення послідовностей символів змінної довжини в потоці символів 307 на кодові слова змінних довжин, які вставляються в потік бітів 312, при цьому, відповідно, кожен з ентропійних кодерів 310 першої підмножини використовує правило бієктивного перетворення, згідно з яким кодові слова головного коду без префіксу з $(2n-1) \geq 3$ кодовими словами перетворюються на кодові слова допоміжного коду без префіксу, який ідентичний головному коду з префіксом так, що усі, але два кодові слова головного коду без префіксу перетворюються на ідентичні кодові слова

50

55

60

допоміжного коду без префіксу, тоді як два кодові слова головного і допоміжного кодів без префіксу мають різні довжини і перетворюються один на інший взаємозамінно, при цьому ентропійні кодери можуть використовувати різне n для охоплення різних частин вищезгаданого інтервалу ймовірності. Перший код без префіксу може формуватися так, що його кодовими словами є $(a,b)_2, (a,a,b)_3, \dots, (a,\dots,a,b)_n, (a,\dots,a)_n, (b,a)_2, (b,b,a)_3, \dots, (b,\dots,b,a)_{n-1}, (b,\dots,b)_{n-1}$, і двома кодовими словами, які взаємозамінно перетворюються одне на інше, є $(a,\dots,a)_n$ і $(b,\dots,b)_{n-1}$ з $b \neq a$ і $a,b \in \{0,1\}$. Однак, альтернативи є реальними.

Іншими словами кожен з першої підмножини ентропійних кодерів може конфігуруватися при перетворенні символів, надісланих до відповідного ентропійного кодера, на відповідний потік бітів для перевірки першого символу, надісланого до відповідного ентропійного кодера, для визначення чи (1) перший символ дорівнює $a \in \{0,1\}$, у випадку чого відповідний ентропійний кодер конфігурується для перевірки наступних символів, надісланих до відповідного ентропійного кодера, для визначення чи (1.1) $b \neq a$ і $b \in \{0,1\}$ серед наступних $n-1$ символів, які слідують після першого символу, у випадку чого відповідний ентропійний кодер конфігурується для запису кодового слова у відповідний потік бітів, який дорівнює першому символу, за яким слідують наступні символи, надіслані до відповідного ентропійного кодера, аж до символу b ; (1.2) чи символ b не трапляється серед наступних $n-1$ символів, які слідують після першого символу, у випадку чого відповідний ентропійний кодер конфігурується для запису кодового слова у відповідний потік бітів, яке дорівнює $(b,\dots,b)_{n-1}$; або (2) чи перший символ дорівнює b , у випадку чого відповідний ентропійний кодер конфігурується для перевірки наступних символів, надісланих до відповідного ентропійного кодера, для визначення (2.1) чи трапляється a серед наступних $n-2$ символів, які слідують після першого символу, у випадку чого відповідний ентропійний кодер конфігурується для запису кодового слова у відповідний потік бітів, який дорівнює першому символу, за яким слідують наступні символи, надіслані до відповідного ентропійного кодера, аж до символу a ; або (2.2) чи не трапляється a серед наступних $n-2$ символів, які слідують після першого символу, у випадку чого відповідний ентропійний кодер конфігурується для запису кодового слова у відповідний потік бітів, яке дорівнює $(a,\dots,a)_n$.

Окрім того, або альтернативно, друга підмножина ентропійних кодерів 10 може бути кодером змінної довжини, сконфігурованим для перетворення послідовностей символів змінних довжин на, відповідно, кодові слова фіксованих довжин, при цьому кожен з ентропійних кодерів другої підмножини використовує правило бієктивного перетворення, згідно з яким кодові слова головного зрізаного унарного коду з 2^n+1 кодовими словами типу $\{(a), (ba), (bba), \dots, (b\dots ba), (bb\dots b)\}$ з $b \neq a$ і $a,b \in \{0,1\}$ перетворюються на кодові слова допоміжного коду без префіксу так, що кодове слово $(bb\dots b)$ головного зрізаного унарного коду перетворюється на кодове слово (c) допоміжного коду без префіксу і усі інші кодові слова $\{(a), (ba), (bba), \dots, (b\dots ba)\}$ головного зрізаного унарного коду перетворюються на кодові слова, які мають (d) з $c \neq d$ і $c,d \in \{0,1\}$ як префікс і n -бітове слово як суфікс, при цьому ентропійні кодери використовують різне n . Кожен з другої підмножини ентропійних кодерів може конфігуруватися так, що n -бітове слово є n -бітовим представленням кількості чисел b у відповідному кодовому слові головного зрізаного унарного коду. Однак, альтернативи є реальними.

Знову, з проєкції режиму роботи відповідного кодера 10 кожен з другої підмножини ентропійних кодерів може конфігуруватися під час перетворення символів, надісланих до відповідного ентропійного кодера, на відповідний потік бітів для підрахунку кількості символів b в послідовності символів, надісланих до відповідного ентропійного кодера, доки не трапиться символ a або доки кількість послідовності символів, надісланих до відповідного ентропійного кодера, не досягне величини 2^n з усіма 2^n символами послідовності, яка є числом b , і (1), якщо кількість символів b дорівнює 2^n , записують c з $c \in \{0,1\}$ як кодове слово допоміжного коду без префіксу у відповідний потік бітів, і (2) якщо кількість символів b менша ніж 2^n , то записують кодове слово допоміжного коду без префіксу у відповідний потік бітів, який має (d) з $c \neq d$ і $d \in \{0,1\}$ як префікс, а n -бітове слово, яке визначається в залежності від кількості символів b , як суфікс.

Також додатково або альтернативно, наперед визначений один з ентропійних кодерів 10 може бути кодером змінної довжини, сконфігурованим для перетворення послідовностей символів фіксованих довжин на, відповідно, кодові слова змінних довжин, при цьому наперед визначений ентропійний кодер використовує правило бієктивного перетворення, згідно з яким 2^3 кодових слова довжиною 3 головного коду перетворюються на кодові слова допоміжного коду без префікса так, що кодове слово $(aaa)_3$ головного коду з $a \in \{0,1\}$ перетворюється на кодове слово (c) з $c \in \{0,1\}$, при цьому усі три кодові слова головного коду, які мають точно одне b з $b \neq a$ і $b \in \{0,1\}$, перетворюються на кодові слова, які мають (d) з $c \neq d$ і $d \in \{0,1\}$ як префікс і відповідне перше 2-бітове слово з першої множини 2-бітових слів як суфікс, при цьому усі три кодові слова головного коду, які мають точно одне a , перетворюються на кодові слова, які мають (d) як

префікс, і конкатенацію першого 2-бітового слова, яке не є елементом першої множини, і другого 2-бітового слова з другої множини 2-бітових слів як суфікс, і при цьому кодове слово $(bbb)_3$ перетворюється на кодове слово, яке має (d) як префікс, і конкатенацію першого 2-бітового слова, яке не є елементом першої множини, і другого 2-бітового слова, яке не є елементом другої множини, як суфікс. Перше 2-бітове слово кодових слів головного коду, яке має точно один символ b, може бути 2-бітовим представленням положення символу b у відповідному кодовому слові головного коду, а друге 2-бітове слово кодових слів головного коду, який має точно один символ a, може бути 2-бітовим представленням положення символу a у відповідному кодовому слові головного коду. Однак, альтернативи є реальними.

Знову, наперед визначений один з ентропійних кодерів може конфігуруватися під час перетворення символів, надісланих до наперед встановленого ентропійного кодера, на відповідний потік бітів, для перевірки символів, які надсилаються до наперед встановленого ентропійного кодера в триплетах, щодо того (1) чи триплет складається з символів a, у випадку чого наперед визначений ентропійний кодер конфігурується для запису кодового слова (c) у відповідний потік бітів, (2) чи триплет точно містить один символ b, у випадку чого наперед визначений ентропійний кодер конфігурується для запису кодового слова, яке має (d) як префікс, а 2-бітове представлення положення символу b в триплеті як суфікс, у відповідний потік бітів; (3) чи триплет точно містить один символ a, у випадку чого наперед визначений ентропійний кодер конфігурується для запису кодового слова, яке має (d) як префікс і конкатенацію першого 2-бітового слова, яке не є елементом першої множини, і 2-бітового представлення положення символу a в триплеті як суфікс, у відповідний потік бітів; або (4) чи триплет складається з символів b, у випадку чого наперед визначений ентропійний кодер конфігурується для запису кодового слова, яке має (d) як префікс і конкатенацію першого 2-бітового слова, яке не є елементом першої множини, і першого 2-бітового слова, яке не є елементом другої множини, як суфікс, у відповідний потік бітів.

Розглядаючи декодувальну сторону, бачимо, що тільки що описані варіанти виконання розкривають декодер для відновлення послідовності символів 326, який містить множину ентропійних декодерів 322, кожен з яких сконфігурований для перетворення відповідного потоку бітів 324 на символи 321; присвоювач 316, сконфігурований для присвоєння ряду параметрів на кожен символ 326 послідовності символів, які відновлюються, на основі інформації, яка міститься в попередньо відновлених символах послідовності символів; і селектор 318, сконфігурований для одержання кожного символу 325 з послідовності символів, які відновлюються, з вибраного одного з множини ентропійних декодерів, при цьому вибір залежить від ряду параметрів, визначених для відповідного символу. Згідно з тільки що описаними варіантами виконання принаймні перша підмножина ентропійних декодерів 322 є декодерами змінної довжини, сконфігурованими для перетворення кодових слів змінних довжин на послідовності символів змінних довжин, при цьому відповідно кожен з ентропійних декодерів 22 першої підмножини використовує правило бієктивного перетворення, згідно з яким кодові слова головного коду без префіксу з $(2n-1) \geq 3$ кодovими словами перетворюються на кодові слова допоміжного коду без префіксу, який ідентичний головному коду з префіксом, так, що усі, але два кодові слова головного коду без префіксу перетворюються на ідентичні кодові слова допоміжного коду без префіксу, тоді як два кодові слова головного і допоміжного коду без префіксу мають різні довжини і перетворюються один на інший взаємозамінно, при цьому ентропійні кодери використовують різні n. Перший код без префіксу може формуватися так, що кодовими словами першого коду без префіксу є $(a,b)_2, (a,a,b)_3, \dots, (a,\dots,a,b)_n, (a,\dots,a)_n, (b,a)_2, (b,b,a)_3, \dots, (b,\dots,b,a)_{n-1}, (b,\dots,b)_{n-1}$, і двома кодовими словами, які перетворюються одне на інше взаємозамінним чином, можуть бути $(a,\dots,a)_{n-1}$ і $(b,\dots,b)_{n-1}$ з $b \neq a$ і $a,b \in \{0,1\}$. Однак, альтернативи є реальними.

Кожен з першої підмножини ентропійних кодерів може конфігуруватися під час перетворення відповідного потоку бітів на символи для перевірки першого біту відповідного потоку бітів для визначення (1) чи перший біт дорівнює 0 $\{0,1\}$, у випадку чого відповідний ентропійний кодер конфігурується для перевірки наступних бітів відповідного бітового потоку для визначення чи $(1.1) b \neq a$ і $b \in \{0,1\}$ трапляється в наступних n-1 бітах, які слідують після першого біта, у випадку чого відповідний ентропійний декодер конфігурується для відновлення послідовності символів, яка дорівнює першому біту, за яким слідують наступні біти відповідного потоку бітів аж до біту b; або (1.2) чи не трапляється біт b серед наступних n-1 бітів, які слідують після першого біта, у випадку чого відповідний ентропійний декодер конфігурується для відновлення послідовності символів, яка дорівнює $(b,\dots,b)_{n-1}$; або (2) чи перший біт дорівнює b, у випадку чого відповідний ентропійний декодер конфігурується для перевірки наступних бітів відповідного потоку бітів для визначення того (2.1) чи трапляється символ a серед наступних n-

2 бітів, які слідують після першого біта, у випадку чого відповідний ентропійний декодер конфігурується для відновлення послідовності символів, яка дорівнює першому біту, за яким слідують наступні біти відповідного бітового потоку аж до символу a ; або (2.2) чи не трапляється символ a серед наступних $n-2$ бітів, які слідують після першого біта, у випадку чого відповідний ентропійний декодер конфігурується для відновлення послідовності символів, яка дорівнює $(a, \dots, a)_n$.

Додатково або альтернативно, принаймні друга підмножина ентропійних декодерів 322 може бути декодером змінної довжини, сконфігурованим для перетворення кодових слів фіксованих довжин на, відповідно, послідовності символів змінних довжин, при цьому кожен з ентропійних декодерів другої підмножини використовує правило бієктивного перетворення, згідно з яким кодові слова допоміжного коду без префіксу перетворюються на кодові слова головного зрізаного унарного коду з 2^n+1 кодovими словами типу $\{(a), (ba), (bba), \dots, (b \dots ba), (bb \dots b)\}$ з $b \neq a$ і $a, b \in \{0,1\}$ так, що кодове слово (c) допоміжного коду без префіксу перетворюється на кодове слово $(bb \dots b)$ головного зрізаного унарного коду і кодові слова, які мають (d) з $c \neq d$ і $c, d \in \{0,1\}$ як префікс і n -бітове слово як суфікс, перетворюються на відповідне одне з інших кодових слів $\{(a), (ba), (bba), \dots, (b \dots ba)\}$ головного зрізаного унарного коду, при цьому ентропійні декодери використовують різні n . Кожен з другої підмножини ентропійних декодерів може конфігуруватися так, що n -бітове слово є n -бітовим представленням кількості символів b у відповідному кодовому слові головного усиченого унарного коду. Однак, альтернативи є реальними.

Кожен з другої підмножини ентропійних декодерів може бути декодером змінної довжини, сконфігурованим для перетворення кодових слів фіксованих довжин на, відповідно, послідовності символів змінних довжин і сконфігурованим під час перетворення потоку бітів відповідного ентропійного декодера на символи для перевірки першого біта відповідного потоку бітів для визначення того чи (1) він дорівнює c з $c \in \{0,1\}$, у випадку чого відповідний ентропійний декодер конфігурується для відновлення послідовності символів, яка дорівнює $(bb \dots b)_2^n$ з $b \in \{0,1\}$; або (2) чи він дорівнює d з $c \neq d$ і $c, d \in \{0,1\}$, у випадку чого відповідний ентропійний декодер конфігурується для визначення n -бітового слова з n подальших бітів відповідного потоку бітів, які слідують після першого біту, і для відновлення з нього послідовності символів, яка має тип $\{(a), (ba), (bba), \dots, (b \dots ba), (bb \dots b)\}$ з $b \neq a$ і $b \in \{0,1\}$ з кількістю символів b , яка залежить від n -бітового слова.

Додатково або альтернативно, наперед визначений один з ентропійних декодерів 322 може бути декодером змінної довжини, сконфігурованим для перетворення кодових слів змінних довжин на, відповідно, послідовності символів фіксованих довжин, при цьому наперед визначений ентропійний декодер використовує правило бієктивного перетворення, згідно з яким кодові слова допоміжного коду без префіксу перетворюються на 2^3 кодові слова довжиною 3 головного коду так, що кодове слово (c) з $c \in \{0,1\}$ перетворюється на кодове слово $(aaa)_3$ головного коду з $a \in \{0,1\}$, при цьому кодові слова, які мають (d) з $c \neq d$ і $d \in \{0,1\}$ як префікс і відповідне перше 2-бітове слово з першої множини трьох 2-бітових слів як суфікс, перетворюються на усі три кодові слова головного коду, який має точно один символ b з $b \neq a$ і $b \in \{0,1\}$, при цьому кодові слова, які мають (d) як префікс і конкатенацію першого 2-бітового слова, яке не є елементом першої множини, і другого 2-бітового слова з другої множини трьох 2-бітових слів як суфікс, перетворюються на усі три кодові слова головного коду, який має точно один символ a , і кодове слово, яке має (d) як префікс і конкатенацію першого 2-бітового слова, яке не є елементом першої множини, і другого 2-бітового слова, яке не є елементом другої множини, як суфікс перетворюється на кодове слово $(bbb)_3$. Перше 2-бітове слово кодових слів головного коду, який має точно один символ b , може бути 2-бітовим представленням положення символу b у відповідному кодовому слові головного коду, а друге 2-бітове слово кодових слів головного коду, який має точно один символ a , може бути 2-бітовим представленням положення символу a у відповідному кодовому слові головного коду. Однак, альтернативи є реальними.

Наперед визначений один з ентропійних декодерів може бути декодером змінної довжини, сконфігурованим для перетворення кодових слів змінних довжин на послідовності символів, кожна з яких містить три символи, відповідно, і під час перетворення потоку бітів відповідного ентропійного декодера на символи сконфігурований для дослідження першого біту відповідного потоку бітів для визначення того факту (1) чи перший біт відповідного потоку бітів дорівнює c з $c \in \{0,1\}$, у випадку чого наперед визначений ентропійний декодер конфігурується для відновлення послідовності символів, яка дорівнює $(aaa)_3$ з $a \in \{0,1\}$, або (2) чи перший біт відповідного потоку бітів дорівнює d з $c \neq d$ і $d \in \{0,1\}$, у випадку чого наперед визначений ентропійний декодер конфігурується для визначення першого 2-бітового слова з 2 подальшими бітами відповідного потоку бітів, які слідують після першого біта, і перевірки першого 2-бітового

слова для визначення того факту (2.1) чи перше 2-бітове слово не є елементом першої множини з трьох 2-бітових слів, у випадку чого наперед визначений ентропійний декодер конфігурується для відновлення послідовності символів, яка має точно один символ b з $b \neq a$ і $b \in \{0,1\}$, при цьому положення символу b у відповідній послідовності символів залежить від першого 2-бітового слова, або (2.2) чи перше 2-бітове слово є елементом першої множини, у випадку чого наперед визначений ентропійний декодер конфігурується для визначення другого 2-бітового слова з 2 подальших бітів відповідного потоку бітів, які слідують після двох бітів, з яких було визначене перше 2-бітове слово, і для перевірки другого 2-бітового слова для визначення того факту (3.1) чи друге 2-бітове слово не є елементом другої множини з трьох 2-бітових слів, у випадку чого наперед визначений ентропійний декодер конфігурується для відновлення послідовності символів, яка має точно один символ a , при цьому положення символу a у відповідній послідовності символів залежить від другого 2-бітового слова, або (3.2) чи друге 2-бітове слово є елементом другої множини з трьох 2-бітових слів, у випадку чого наперед визначений ентропійний декодер конфігурується для відновлення послідовності символів, яка дорівнює $(bbb)_3$.

Тепер, після опису загальної концепції схеми кодування відеоданих, варіанти виконання представленого винаходу описуються відносно вищезгаданих варіантів виконання. Іншими словами, охарактеризовані нижче варіанти виконання можуть втілюватися шляхом використання вищенаведених схем і, навпаки, вищезгадані схеми кодування можуть втілюватися з використанням охарактеризованих нижче варіантів виконання.

У вищенаведених варіантах виконання, описаних відносно Фіг. 7-9, ентропійний кодер і декодери з Фіг. 1 - 6 втілювалися у відповідності з концепцією PIPE (ентропія поділу інтервалу ймовірності). Один спеціальний варіант виконання використовує арифметичні кодери/декодери 310 і 322 з єдиним станом ймовірності. Як буде описуватися нижче, у відповідності з альтернативним варіантом виконання, об'єкти 306-310 і відповідні об'єкти 318 - 322 можуть замінятися спільним ентропійним кодувальним засобом. Як приклад, уявіть арифметичний кодувальний засіб, який просто керує одним спільним станом R і L , і кодує усі символи з одержанням одного спільного потоку бітів, таким чином відмовляючись від вигідних аспектів представленої концепції PIPE, що стосується паралельної обробки, але уникаючи необхідності чергування часткових потоків

бітів, як обговорюється нижче. Роблячи це, кількість станів ймовірності, завдяки яким ймовірності контексту оцінюються шляхом оновлення даних (таких як таблиця пошуку), може бути більшою за кількість станів ймовірності, завдяки яким виконується підрозбиття інтервалу ймовірності. Тобто, аналогічно до дискретизації величина ширини інтервалу ймовірності перед індексацією в таблиці R_{tab} , а також індекс стану ймовірності можуть дискретизуватися. Вищенаведений опис можливого втілення для окремих кодерів/декодерів 310 і 322 може, таким чином, поширюватися, наприклад, на втілення ентропійних кодерів/декодерів 318-322/306-310 як адаптивних до контексту двійкових арифметичних кодувальних/декодувальних засобів:

Будучи більш точними, у відповідності з варіантом виконання, ентропійний кодер, підключений до виходу присвоювача параметрів (який функціонує тут як контекстний присвоювач), може працювати наступним чином:

0. Присвоювач 304 надсилає значення інформаційної величини разом з параметром ймовірності. Ймовірністю є $pState_current[bin]$.

1. Таким чином, ентропійний кодувальний засіб приймає: 1) $valLPS$, 2) інформаційну величину і 3) оцінку розподілу ймовірності $pState_current[bin]$. $pState_current[bin]$ може мати більше станів, ніж кількість розрізняваних індексів стану ймовірності R_{tab} . Якщо так, то $pState_current[bin]$ може дискретизуватися як, наприклад, шляхом ігнорування m LSBs з m , більшим за або рівним 1 і переважно рівним 2 або 3, для одержання p_state , тобто індексу, який потім використовується для доступу до таблиці R_{tab} . Однак, дискретизація може не виконуватися, тобто. p_state може дорівнювати $pState_current[bin]$.

2. Потім, виконується дискретизація R (Як згадується вище: будь-який один R (і відповідний L з одним спільним потоком бітів) використовується/контролюється для усіх розрізняваних величин p_state або один R (і відповідний L з відповідним частковим потоком бітів на пару R/L) на розрізнявану величину p_state , яка в останньому випадку повинна відповідати наявності одного двійкового кодера 310 на таку величину)

$q_index = Q_{tab}[R \gg q]$ (або деяка інша форма дискретизації).

3. Потім, виконується визначення R_{LPS} і R :

$R_{LPS} = R_{tab}[p_state][q_index]$; R_{tab} зберегла попередньо обраховані величини для $p[p_state] \cdot Q[q_index]$

$R = R - R_{LPS}$ [тобто, R попередньо оновлюється якщо б "елемент вибірки" був MPS]

4. Обрахунок нового часткового інтервалу:

if (bin=1- valMPS) then

L←L+R

R←R_{LPS}

5. Повторна нормалізація L і R, запис бітів,

Аналогічно, ентропійний декодер, підключений до виходу присвоювача параметрів (який функціонує тут як контекстний присвоювач), може працювати наступним чином:

0. Присвоювач 304 надсилає значення інформаційної величини разом з параметром ймовірності. Ймовірність дорівнює pState_current[bin].

1. Таким чином, ентропійний кодувальний засіб приймає запит на надання інформаційної величини разом з: 1) valLPS, і 2) оцінкою розподілу ймовірності pState_current[bin]. pState_current[bin] може мати більше станів ніж кількість розрізняваних індексів станів ймовірності таблиці Rtab. Якщо так, то pState_current[bin] може дискретизуватися, наприклад, шляхом ігнорування m LSBs з m, яке більше за або рівне 1 і переважно дорівнює 2 або 3, для одержання p_state, тобто індексу, який потім використовується для доступу до таблиці Rtab. Дискретизація може, однак, не виконуватися, тобто, p_state може дорівнювати pState_current[bin].

2. Потім, виконується дискретизація R (як згадано вище: будь-який один R (і відповідний V з одним спільним потоком бітів) використовується/контролюється для усіх розрізняваних величин p_state, або один R (і відповідний V з відповідним частковим потоком бітів на пару R/L) на розрізнявану величину p_state, яка в останньому випадку повинна відповідати наявності одного двійкового кодера 310 на таку величину)

q_index=Qtab[R>>q] (або деяка інша форма дискретизації)

3. Потім, виконують визначення R_{LPS} і R:

R_{LPS}=Rtab[p_state][q_index]; Rtab зберегла попередньо обраховані величини для p[p_state]Q[q_index]

R=R-R_{LPS} [тобто, R попередньо оновлюється, якщо "інформаційною величиною" було MPS]

4. Визначення інформаційної величини, яке залежить від положення часткового інтервалу:

if (V³ R) then

bin←1-valMPS (інформаційна величина декодується як LPS; двійковий буферний селектор 18 буде одержувати реальне значення інформаційної величини шляхом використання цієї інформації про інформаційну величину і valMPS)

V←V-R

R←RLPS

або

bin←valMPS (інформаційна величина декодується як MPS;

реальне значення інформаційної величини одержується шляхом використання цієї інформації про інформаційну величину та valMPS)

5. Повторна нормалізація R, зчитування одного біта і оновлення V,

Як описано вище, присвоювач 4 присвоює pState_current[bin] кожній інформаційній величині. Зв'язування може виконуватися на основі вибору контексту. Тобто, присвоювач 4 може вибирати контекст, використовуючи індекс контексту ctxldx, який, у свою чергу, має відповідну зв'язану з ним ймовірність pState_current. Оновлення ймовірності може виконуватися кожен раз, коли ймовірність pState_current[bin] використана для поточної інформаційної величини. Оновлення стану ймовірності pState_current[bin] виконується в залежності від значення кодованого біту:

if (bit=1-valMPS) then

pState_current ← Next_State_LPS [pState_current]

if (pState_current=0) then valMPS ← 1-valMPS

else

pState_current ← Next_State_MPS [pState_current]

Якщо надається більше ніж один контекст, то адаптація виконується контекстно, тобто, pState_current[ctxldx] використовується для кодування і потім для оновлення з використанням поточного значення інформаційної величини (кодованої або, відповідно, декодованої).

Як буде описано детальніше нижче, у відповідності з описаними тепер варіантами виконання кодер і декодер можуть необов'язково втілюватися для роботи в різних режимах, а саме: в режимі низької складності (LC) і вискоєфективному режимі (HE). Це головним чином ілюструється далі відносно до PIPE кодування (потім згадуючи режими LC і HE PIPE), але опис деталей складності масштабування легко переноситься на інші втілення засобів ентропійного

кодування/декодування, таких як варіант виконання з використанням одного спільного адаптивного до контексту арифметичного кодера/декодера.

У відповідності з описаними нижче варіантами виконання обидва режими ентропійного кодування можуть використовувати

5 - однаковий синтаксис і семантику (для послідовності синтаксичних елементів 301 і, відповідно, 327)

- однакові схеми бінаризації для усіх синтаксичних елементів (як на даний момент специфікується для CABAC) (тобто, бінаризатори можуть працювати незалежно від активованого режиму)

10 - використання одних і тих же PIPE кодів (тобто, двійкові кодери/декодери можуть працювати незалежно від активованого режиму)

- використання 8 бітових ініціалізуючих величин моделі ймовірності (замість 16 бітових ініціалізуючих величин, як на даний момент специфіковано для CABAC)

15 Загалом кажучи, LC-PIPE відрізняється від HE-PIPE складністю обробки, як, наприклад, складність вибору PIPE доріжки 312 для кожної інформаційної величини.

Наприклад, режим низької складності (LC) може функціонувати за наступних обмежень: для кожної інформаційної величини (binIdx) може існувати точно одна модель ймовірності, тобто, одне значення ctxIdx. Тобто, може не передбачатися вибір/адаптація контексту в LC PIPE. Спеціальні синтаксичні елементи, такі як ті, що використовуються для залишкового кодування, 20 можуть, однак, кодуватися з використанням контекстів, як далі описано нижче. Більше того, усі моделі ймовірності можуть бути неадаптивними, тобто, усі моделі можуть ініціалізуватися на початку кожної частини інформації відповідними модельованими ймовірностями (в залежності від вибору типу частини інформації і QP частини інформації) і можуть зберігатися фіксованими протягом усього процесу обробки частини інформації. Наприклад, можуть підтримуватися тільки 25 8 різних модельованих ймовірностей, які відповідають 8 різним PIPE кодам 310/322, для моделювання контексту і кодування. Спеціальні синтаксичні елементи для залишкового кодування, тобто, significance_coeff_flag і coeff_abs_level_greaterX (з X=1,2), семантика яких описана детальніше нижче, можуть присвоюватися моделям ймовірності так, що (принаймні) групи, наприклад, з 4 синтаксичних елементів кодуються/декодуються однією і тією ж 30 модельованою ймовірністю. Порівняно з CAVLC режим LC-PIPE досягає приблизно тих же характеристик R-D і тієї ж продуктивності.

HE-PIPE може конфігуруватися концептуально подібно до CABAC стандарту H.264 з наступними відмінностями: Двійкове арифметичне кодування (BAC) замінюється PIPE кодуванням (теж саме що й у випадку LC-PIPE). Кожна модель ймовірності, тобто, кожен ctxIdx, 35 може представлятися pipelIdx і refinelIdx, де pipelIdx із значеннями в інтервалі 0...7 представляє модельовану ймовірність 8 різних PIPE кодів. Ця зміна впливає тільки на внутрішнє представлення станів, а не на поведінку самої машини стану (тобто, оцінку ймовірності). Як буде детальніше описано нижче, ініціалізація моделей ймовірності може використовувати 8 бітові ініціалізуючі величини, як вказано вище. Зворотне сканування синтаксичних елементів 40 coeff_abs_level_greaterX (з X=1,2), coeff_abslevel_minus3 і coeff_sign_flag (семантика яких стане яснішою з нижченаведеного обговорення) може виконуватися вздовж тієї ж доріжки сканування, що й пряме сканування (використовуване, наприклад, в кодування карти значущості). Може також спрощуватися одержання контексту для кодування coeff_abs_level_greaterX (with X=1,2). Порівняно з CABAC запропонований HE-PIPE приблизно досягає тих же характеристик R-D з 45 кращою продуктивністю.

Легко побачити, що тільки що згадані режими легко генеруються шляхом виконання, наприклад, вищезгаданого адаптивного до контексту двійкового арифметичного кодувального/декодувального засобу так, що він працює в різних режимах.

Таким чином, у відповідності з варіантом виконання у відповідності з першим аспектом 50 представленого винаходу декодер для декодування потоку даних може формуватися, як зображено на Фіг. 11. Декодер передбачений для декодування потоку даних 401, такого як поперемінний потік бітів 340, в який кодуються мультимедійні дані, такі як відеодані. Декодер містить перемикач режимів 400, сконфігурований для активування режиму низької складності або вискоєфективного режиму в залежності від потоку даних 401. Для цього, потік даних 401 55 може містити синтаксичний елемент, такий як двійковий синтаксичний елемент, який має двійкову величину 1 у випадку режиму низької складності, який активується, і має двійкову величину 0 у випадку вискоєфективного режиму, який активується. Очевидно, зв'язок між інформаційною величиною і режимом кодування може перемикатися, а недвійковий синтаксичний елемент, який має більше ніж два можливі значення, міг би також 60 використовуватися. Оскільки реальний вибір між обома режимами все ще нечіткий до прийому

відповідного синтаксичного елемента, то цей синтаксичний елемент може міститися в деякому ведучому заголовку кодованого потоку даних 401, наприклад, з фіксованою оцінкою ймовірності або моделлю ймовірності, або записуватися в потік даних 401, тобто, коли використовується байпасний режим.

5 Окрім того, декодер з Фіг. 11 містить множину ентропійних декодерів 322, кожен з яких сконфігурований для перетворення кодових слів в потоці даних 401 на часткові послідовності 321 символів. Як описано вище, обернений перемешувач 404 може підключатися між входами ентропійних декодерів 322, з одного боку, і входом декодера з Фіг. 11, де використовується потік даних 401, з іншого боку. Окрім того, як вже описувалося вище, кожен з ентропійних декодерів 10 322 може зв'язуватися з відповідним інтервалом ймовірності, при цьому інтервали ймовірності різних ентропійних декодерів разом охоплюють увесь інтервал ймовірності від 0 до 1 або від 0 до 0,5 у випадку ентропійних декодерів 322, які мають справу з MPS і LPS скоріше ніж з абсолютними символічними величинами. Деталі, які стосуються цього питання, були описані вище. Пізніше припускається, що кількість декодерів 322 становить 8 з PIPE індексом, який 15 присвоюється кожному декодеру, але будь-яка інша кількість також можлива. Окрім того, один з цих кодерів (далі це ілюстративно той кодер, який має `pipe_id` 0) оптимізується для інформаційних величин, які мають рівноцінну статистику, тобто, їх двійкова величина рівноцінно припускає 1 і 0. Цей декодер може просто проходитися по інформаційних величинах. Відповідний кодер 310 працює так само. Навіть будь-яка маніпуляція з інформаційною 20 величиною, яка залежить від значення найбільш ймовірної інформаційної величини, `valMPS`, за допомогою селекторів 402 і, відповідно, 502 може усуватися. Іншими словами, ентропія відповідного часткового потоку є вже оптимальною.

Окрім того, декодер з Фіг. 11 містить селектор 402, сконфігурований для одержання кожного символу послідовності 326 символів з вибраного одного з множини ентропійних декодерів 322. 25 Як згадано вище, селектор 402 може ділитися на присвоювач параметрів 316 і селектор 318. Десимволізатор 314 конфігурується для десимволізації послідовності 326 символів для одержання послідовності 327 синтаксичних елементів. Реконструктор 404 конфігурується для відновлення мультимедійних даних 405 на основі послідовності синтаксичних елементів 327. Селектор 402 конфігурується для виконання вибору, який залежить від одного активованого 30 режиму, вибраного серед режиму низької складності і високоефективного режиму, як вказано стрілкою 406.

Як вже відзначено вище, реконструктор 404 може бути частиною блочного прогнозувального відеодекодера, який працює на фіксованому синтаксисі і семантиці синтаксичних елементів, тобто, фіксований відносно вибору режиму за допомогою перемикача режимів 400. Тобто, 35 структура реконструктора 404 не страждає від здатності перемикати режими. Точніше, реконструктор 404 не збільшує надмірне втілення завдяки здатності перемикання режимів, яке забезпечується перемикачем режимів 400 і принаймні функціональністю стосовно залишкових даних, і дані прогнозування залишаються однаковими незалежно від режиму, вибраного перемикачем 400. Те ж саме застосовується, однак, стосовно ентропійних декодерів 322. Усі ці 40 декодери 322 повторно використовуються в обох режимах і, відповідно, відсутні додаткові втілення, хоча декодер з Фіг. 11 сумісний з обома режимами: режимом низької складності і високоефективним режимом.

Як побічний аспект слід відзначити, що декодер з Фіг. 11 не тільки здатен працювати на незалежних потоках даних або в одному режимі або в іншому режимі. Скоріше, декодер з Фіг. 45 11, а також потік даних 401 можуть конфігуруватися так, що перемикання між обома режимами повинно бути навіть можливим під час обробки однієї частини мультимедійних даних, як, наприклад під час обробки відеоданих або деякої частини аудіоінформації, для, наприклад, керування складністю кодування на кодувальній стороні в залежності від зовнішніх або навколишніх умов, таких як стан акумулятора або подібне, з використанням каналу зворотного зв'язку від декодера до кодера, відповідно, для керування в замкненому контурі вибором 50 режиму.

Таким чином, декодер з Фіг. 11 працює подібно в обох випадках: у випадку режиму низької складності (LC), який вибирається, або високоефективного режиму (HE), який вибирається. Реконструктор 404 виконує відновлення з використанням синтаксичних елементів і запрошує 55 поточний синтаксичний елемент наперед визначеного типу шляхом обробки або задоволення деякої рекомендації щодо синтаксичної структури. Десимволізатор 314 запрошує ряд інформаційних величин для виконання дійсної бінаризації для синтаксичного елемента, запрошеного реконструктором 404. Очевидно, що у випадку двійкового алфавіту, бінаризація, виконувана десимволізатором 314, зводиться просто до подачі відповідної інформаційної

величини/символа 326 до реконструктора 404 як на даний момент запрошений двійковий синтаксичний елемент.

Однак, селектор 402 функціонує незалежно в режимі, вибраному перемикачем режимів 400. Режим роботи селектора 402 має тенденцію бути складнішим у випадку високоефективного режиму і менш складним у випадку режиму низької складності. Більше того, наступне обговорення покаже, що режим роботи селектора 402 в режимі нижчої складності також має тенденцію знижувати швидкість, з якою селектор 402 змінює вибір серед ентропійних декодерів 322 при одержанні послідовних символів з ентропійних декодерів 322. Іншими словами, в режимі низької складності існує вища ймовірність, що безпосередньо розташовані один за одним послідовні символи одержуються з одного і того ж ентропійного декодера серед множини ентропійних декодерів 322. Це, у свою чергу, дозволяє швидше одержання символів з ентропійних декодерів 322. У високоефективному режимі, у свою чергу, режим роботи селектора 402 має тенденцію до досягання вибору серед ентропійних декодерів 322, де інтервал ймовірності, зв'язаний з відповідним вибраним ентропійним декодером 322, більш близько відповідає реальній символічній статистиці символу, на даний момент одержаного селектором 402, таким чином досягаючи кращого коефіцієнта стиснення на кодувальній стороні при генеруванні відповідного потоку даних у відповідності з високоефективним режимом.

Наприклад, різна поведінка селектора 402 в обох режимах може реалізовуватися наступним чином. Наприклад, селектор 402 може конфігуруватися для виконання для наперед визначеного символу вибору серед множини ентропійних декодерів 322, який залежить від попередньо одержаних символів послідовності 326 символів у випадку високоефективного режиму роботи, який активується, і незалежно від будь-яких попередньо одержаних символів послідовності символів у випадку режиму низької складності, який активується. Залежність від попередньо одержаних символів послідовності 326 символів може впливати з адаптивності до контексту і/або адаптивності до ймовірності. Обидві адаптивності можуть деактивуватися в селекторі 402 під час режиму низької складності.

У відповідності з подальшим варіантом виконання потік даних 401 може структуруватися на послідовні частини, такі як частини масиву даних, кадри, група картинок, послідовності кадрів або подібне, і кожен символ послідовності символів може зв'язуватися з відповідним одним з множини типів символів. У цьому випадку селектор 402 може конфігуруватися для варіювання для символів наперед визначеного типу в поточній частині масиву даних, при цьому вибір залежить від попередньо одержаних символів послідовності символів наперед визначеного типу в поточній частині масиву даних у випадку високоефективного режиму, який активується, і для полишення вибору сталим в поточній частині у випадку режиму низької складності, який активується. Тобто, селектору 402 можуть надавати можливість змінювати вибір серед ентропійних декодерів 322 для наперед встановленого типу символу, але ці зміни обмежуються появою між переходами між послідовними частинами масиву даних. Завдяки цьому заходу оцінки реальної статистики символів обмежуються рідкими моментами часу, тоді як складність кодування зменшується протягом більшості часу.

Окрім того, кожен символ послідовності 326 символів може зв'язуватися з відповідним одним з множини типів символів, а селектор 402 може конфігуруватися для наперед визначеного символу наперед визначеного типу для вибору одного з множини контекстів, який залежить від попередньо одержаних символів послідовності 326 символів, і виконання вибору серед ентропійних декодерів 322, який залежить від моделі ймовірності, зв'язаної з вибраним контекстом разом з оновленням моделі ймовірності, зв'язаної з вибраним контекстом, який залежить від наперед визначеного символу у випадку високоефективного режиму, який активується, і для виконання вибору одного з множини контекстів, який залежить від попередньо одержаних символів послідовності 326 символів, і для виконання вибору серед ентропійних декодерів 322, який залежить від моделі ймовірності, зв'язаної з вибраним контекстом, разом з полишенням моделі ймовірності, зв'язаної з вибраним контекстом, сталою у випадку режиму низької складності, який активується. Тобто, селектор 402 може використовувати адаптивність до контексту відносно певного типу синтаксичного елемента в обох режимах з одночасним уникненням адаптації до ймовірності у випадку режиму низької складності (LC).

Альтернативно, замість повного усунення адаптації до ймовірності, селектор 402 може просто знижувати швидкість оновлення адаптації до ймовірності режиму низької складності (LC) відносно високоефективного (HE) режиму.

Окрім того, можливі спеціальні LC-ріпе аспекти, тобто, аспекти режиму низької складності (LC), можна описати іншими словами. Зокрема, неадаптивні моделі ймовірності могли б використовуватися в режимі низької складності (LC). Неадаптивна модель ймовірності може або

мати жорстко закодовану, тобто загалом стали ймовірність, або її ймовірність зберігається фіксованою протягом тільки обробки частини масиву даних і, таким чином, може визначатися залежно від типу частини масиву даних і QP, тобто, від параметра дискретизації, який, наприклад, сигналізується в потоці даних 401 для кожної частини масиву даних. Припускаючи, що послідовні інформаційні величини, присвоєні одному і тому ж контексту, дотримуються моделі фіксованої ймовірності, можна декодувати декілька з таких інформаційних величин за один етап, оскільки вони кодуються з використанням одного і того ж ріре коду, тобто, використовуючи однаковий ентропійний декодер, і уникається оновлення ймовірності після кожної декодованої інформаційної величини. Уникнення оновлень ймовірності зберігає операції під час процесу кодування і декодування і, таким чином, також веде до знижень складності і значного спрощення архітектури апаратних засобів.

Неадаптивне обмеження може послаблюватися для усіх або деяких вибраних моделей ймовірності у такий спосіб, що оновлення ймовірності дозволяються після певної кількості інформаційних величин, кодованих/декодованих з використанням цієї моделі. Належний інтервал оновлення дозволяє адаптацію ймовірності з наявністю здатності декодувати за один раз декілька інформаційних величин.

Далі представляється детальніший опис можливих спільних аспектів з масштабованою складністю режимів LC-ріре і HE-ріре. Зокрема, далі описуються аспекти, які можуть використовуватися для режиму LC-ріре і режиму HE-ріре у той же спосіб або у спосіб з масштабованою складністю. Масштабована складність означає, що випадок LC одержується з випадку HE шляхом видалення окремих частин або шляхом заміни їх дещо менш складними частинами. Однак, перед виконанням цього, слід відзначити, що варіант виконання з Фіг. 11 легко перетворюється на вищезгаданий варіант виконання адаптивного до контексту двійкового арифметичного кодування/декодування: селектор 402 і ентропійні декодери 322 повинні об'єднуватися в адаптивний до контексту двійковий арифметичний декодер, який повинен безпосередньо приймати потік даних 401 і вибирати контекст для інформаційної величини, який на даний момент одержується з потоку даних. Це особливо справедливо для адаптивності до контексту і/або адаптивності до ймовірності. Обидві функції/адаптивності можуть деактивуватися або розроблятися більш послабленими під час режиму низької складності.

Наприклад, при втіленні варіанта виконання з Фіг. 11, фаза ентропійного ріре кодування, яка використовує ентропійні декодери 322, могла б використовувати вісім систематичних кодів, які відображають перетворення послідовності символів змінної довжини на послідовність бітів змінної довжини, тобто, кожен ентропійний декодер 322 міг би бути декодером типу v2v, який описаний вище. Концепція PIPE кодування, яка використовує систематичні v2v-коди, спрощується шляхом обмеження кількості Y2Y-кодів. У випадку адаптивного до контексту двійкового арифметичного декодера вона могла б керувати тими ж станами ймовірності для різних контекстів і використовувати ту ж або її дискретизовану версію для підрозбиття інтервалу ймовірності. Перетворення CABAC або станів моделі ймовірності, тобто станів, використовуваних для оновлення ймовірності, на PIPE індекси або індекси ймовірності для пошуку в таблиці Rtab можуть бути тими, що зображені в Таблиці А.

Таблиця А:

Перетворення станів CABAC на PIPE індекси

Стан CABAC	PIPE Індекс	Стан CABAC	PIPE Індекс
0	0	32	5
1		33	
2		34	
3	1	35	
4		36	
5		37	
6		38	
7		39	
8		40	
9		41	
10	2	42	
11		43	
12		44	

13		45	
14		46	6
15	3	47	
16		48	
17		49	
18		50	
19		51	
20		52	
21		53	
22	4	54	
23		55	
24		56	
25		57	
26		58	
27		59	
28		60	
29		61	
30		62	7
31			

Ця модифікована схема кодування може використовуватися як основа для наближення кодування відеоданих з масштабованою складністю. При виконанні адаптації режиму ймовірності селектор 402 або адаптивний до контексту двійковий арифметичний декодер, відповідно, повинен вибирати PIPE декодер 322, тобто, одержувати використовуваний ріре індекс і індекс ймовірності в Rtab, відповідно, на основі індексу стану ймовірності (тут ілюстративний інтервал від 0 до 62) зв'язаний з поточним символом, який декодується, як, наприклад, за допомогою контексту, з використанням перетворення, вказаного в таблиці А, і повинен оновлювати цей індекс стану ймовірності в залежності від на даний момент декодованого символу з використанням, наприклад, спеціальних перехідних величин таблиці, які вказують на індекс наступного стану ймовірності, який переглядається у випадку MPS і, відповідно, LPS. У випадку режиму низької складності (LC), оновлення останнього можна було б уникнути. Навіть перетворення можна було б уникнути у випадку моделей глобально фіксованої ймовірності.

Однак, довільний ентропійний кодувальний пристрій міг би використовуватися і технології у цьому документі можна також використовувати з незначними адаптаціями.

Вищенаведений опис Фіг. 11 скоріше головним чином належить до синтаксичних елементів і типів синтаксичних елементів. Далі описується кодування з конфігурованою складністю рівнів значущості коефіцієнта перетворення.

Наприклад, реконструктор 404 може конфігуруватися для відновлення блока перетворення 200 рівнів значущості коефіцієнта перетворення 202 на основі частини послідовності синтаксичних елементів незалежно від високоефективного режиму або режиму низької складності, який активується, на основі частини послідовності 327 синтаксичних елементів, яка містить непоперемінним способом синтаксичні елементи карти значущості, які визначають карту значущості, яка вказує положення ненульових рівнів коефіцієнта перетворення в блоці 200 перетворення, а потім (після) синтаксичні елементи рівня значущості, які визначають ненульові рівні коефіцієнта перетворення. Зокрема, можуть використовуватися наступні елементи: синтаксичні елементи кінцевого положення (last_significant_pos_x, last_significant_pos_y), які вказують положення останнього ненульового рівня коефіцієнта перетворення в блоці перетворення; перші синтаксичні елементи (coeff_significant_flag), які разом визначають карту значущості і вказують для кожного положення вздовж одномірної доріжки (274), яка веде від положення DC до положення останнього ненульового рівня коефіцієнта перетворення в блоці (200) перетворення, чи рівень коефіцієнта перетворення у відповідному положенні є ненульовим чи ні; другі синтаксичні елементи (coeff_abs_greater1), які вказують для кожного положення одномірної доріжки (274), де, згідно з першими двійковими синтаксичними елементами, розташований ненульовий рівень коефіцієнта перетворення, чи рівень коефіцієнта перетворення у відповідному положенні більший за одиницю; і треті синтаксичні елементи (coeff_abs_greater2, coeff_abs_minus3), які вказують для кожного положення одномірної доріжки, де, згідно з першими двійковими синтаксичними елементами, розташований рівень коефіцієнта перетворення, більший за одиницю, величину, на яку відповідний рівень коефіцієнта перетворення у відповідному положенні перевищує одиницю.

Порядок серед синтаксичних елементів кінцевого положення, перших, других і третіх синтаксичних елементів може бути однаковим для високоефективного режиму і режиму низької складності, і селектор 402 може конфігуруватися для виконання вибору серед ентропійних декодерів 322 для символів, з яких десимволізатор 314 одержує синтаксичні елементи кінцевого положення, перші синтаксичні елементи, другі синтаксичні елементи і/або треті синтаксичні елементи, які по різному залежать від режиму низької складності або високоефективного режиму, який активується.

Зокрема, селектор 402 може конфігуруватися для символів наперед визначеного типу серед послідовності символів, з яких десимволізатор 314 одержує перші синтаксичні елементи і другі синтаксичні елементи, для вибору для кожного символу наперед визначеного типу одного з множини контекстів, який залежить від попередньо одержаних символів наперед визначеного типу серед послідовності символів і для виконання вибору, який залежить від моделі ймовірності, зв'язаної з вибраним контекстом у випадку високоефективного режиму, який активується, і для виконання вибору кусково сталим способом так, що вибір є сталим на послідовних неперервних субчастинах послідовності у випадку режиму низької складності, який активується. Як описано вище, субчастини можуть визначатися в ряді положень, по яких проходять відповідна субчастина вздовж одномірної доріжки 274, або в ряді синтаксичних елементів відповідного типу, вже кодованого поточним контекстом. Тобто, двійкові синтаксичні елементи `coeff_significant_flag`, `coeff_abs_greater1` і `coeff_abs_greater2`, наприклад, кодуються адаптивно до контексту вибором декодера 322 на основі моделі ймовірності вибраного контексту у високоефективному режимі (HE). Також використовується адаптація ймовірності. В режимі низької складності (LC) існують також різні контексти, які використовуються для кожного з двійкових синтаксичних елементів `coeff_significant_flag`, `coeff_abs_greater1` і `coeff_abs_greater2`. Однак, для кожного з цих синтаксичних елементів контекст зберігається статичним для першої частини вздовж доріжки 274 з простою зміною контексту при переході до наступної частини, яка безпосередньо слідує за нею вздовж доріжки 274. Наприклад, кожна частина може займати 4, 8, 16 положень блока 200 в довжину незалежно від того, чи присутній, чи ні для відповідного положення відповідний синтаксичний елемент. Наприклад, `coeff_abs_greater1` і `coeff_abs_greater2` просто присутні для значущих положень, тобто, положень, де або для яких `coeff_significant_flag` становить 1. Альтернативно, кожна частина може становити в довжину 4, 8, 16 синтаксичних елементів незалежно від того, чи, таким чином одержувана відповідна частина охоплює більшу кількість положень блоків. Наприклад, `coeff_abs_greater1` і `coeff_abs_greater2` просто присутні для значущих положень і, таким чином, частини чотирьох синтаксичних елементів можуть охоплювати більше ніж 4 положення блока завдяки положенням між ними вздовж доріжки 274, для якої синтаксичний елемент не передається так, що відсутні `coeff_abs_greater1` і `coeff_abs_greater2`, оскільки відповідний рівень у цьому положенні дорівнює нулю.

Селектор 402 може конфігуруватися для символів наперед визначеного типу серед послідовності символів, з яких десимволізатор одержує перші синтаксичні елементи і другі синтаксичні елементи, для вибору для кожного символу наперед визначеного типу одного з множини контекстів в залежності від кількості попередньо одержаних символів наперед визначеного типу в підпослідовності символів, які мають наперед визначену величину і належать до однієї і тієї ж субчастини, або від кількості попередньо одержаних символів наперед визначеного типу в послідовності символів, які належать одній і тій же субчастині. Перша альтернатива була справедливою для `coeff_abs_greater1`, а допоміжна альтернатива була справедливою для `coeff_abs_greater2` у відповідності з вищенаведеними спеціальними варіантами виконання.

Окрім того, треті синтаксичні елементи, які вказують для кожного положення одномірної доріжки, де, згідно з першими двійковими синтаксичними елементами, розташований рівень значущості коефіцієнта перетворення, більший за одиницю, величини, на яку відповідний рівень значущості коефіцієнта перетворення у відповідному положенні перевищує одиницю, можуть включати цілі синтаксичні елементи, тобто, `coeff_abs_minus3`, і десимволізатор 314 може конфігуруватися для використання функції перетворення, яка керується контрольним параметром, для перетворення області слів символної послідовності на кообласть цілих синтаксичних елементів, і для визначення контрольного параметра на цілий синтаксичний елемент в залежності від цілих синтаксичних елементів попередніх третіх синтаксичних елементів, якщо активується високоефективний режим, і для виконання визначення параметрів кусково сталим чином так, що визначення параметрів є сталим на послідовних неперервних субчастинах підпослідовності у випадку режиму низької складності, який активується, при цьому селектор 402 може конфігуруватися для вибору наперед визначеного одного з ентропійних

декодерів (322) для символів слів символічної послідовності, перетворених на цілі синтаксичні елементи, який зв'язується з рівномірним розподілом ймовірності як у високоефективному режимі так і у режимі низької складності. Тобто, навіть десимволізатор може працювати в залежності від режиму, вибраного перемикачем режимів 400, зображеним пунктирною лінією

407. Замість кусковосталого визначення контрольного параметра десимволізатор 314 може зберігати контрольний параметр сталим під час поточної частини масиву, наприклад, або глобально сталим в часі.

Далі описується моделювання контексту з масштабованою складністю.

Оцінка того ж синтаксичного елемента верхнього лівого сусіднього блока для одержання індексу моделі контексту є спільним наближенням і часто використовується у випадку високоефективного режиму (HE), наприклад, для синтаксичного елемента різниці векторів руху. Однак, ця оцінка вимагає більшого зберігального буфера і не дозволяє пряме кодування синтаксичного елемента. Також, для досягання вищих характеристик кодування, можуть оцінюватися більш доступні сусідні елементи.

В переважному варіанті виконання уся фаза моделювання контексту, яка оцінює синтаксичні елементи сусідніх квадратних або прямокутних блоків або прогнозувальних блоків, присвоюються одній моделі контексту. Це дорівнює деактивуванню адаптивності на фазі вибору моделі контексту. Для такого переважного варіанта виконання вибір моделі контексту, який залежить від двійкового індексу рядка інформаційних величин після бінаризації не модифікується порівняно з поточною структурою для CABAC. В іншому переважному варіанті виконання на додаток до моделі фіксованого контексту для синтаксичних елементів використовують оцінку сусідніх блоків, також фіксують модель контексту для різних двійкових індексів. Відзначаємо, що опис не включає бінаризацію і вибір моделі контексту для різниці векторів руху і синтаксичні елементи, які стосуються кодування рівнів значущості коефіцієнта перетворення.

В переважному варіанті виконання дозволяється тільки оцінка лівого сусіднього блока. Це веде до зменшеного буфера в ланцюзі обробки, оскільки останній блок або кодувальну блочну лінію не потрібно більше зберігати. В подальшому переважному варіанті виконання оцінюються тільки сусідні блоки, які лежать в одному і тому ж кодувальному засобі.

В переважному варіанті виконання оцінюються усі доступні сусідні блоки. Наприклад, на додаток до верхнього лівого сусіднього блока, верхній лівий, верхній правий і нижній правий блок оцінюються у випадку доступності.

Тобто, селектор 402 з Фіг. 11 може конфігуруватися для використання для наперед визначеного символу, який належить до наперед визначеного блока мультимедійних даних, попередньо одержаних символів послідовності символів, яка належить до більшої кількості різних сусідніх блоків мультимедійних даних у випадку високоефективного режиму, який активується, для вибору одного з множини контекстів і для виконання вибору серед ентропійних декодерів 322, який залежить від моделі ймовірності, зв'язаної з вибраним контекстом. Тобто, сусідні блоки можуть бути сусідніми в часовій і/або просторовій області. Просторово сусідні блоки видимі, наприклад, на Фіг. 1 - 3. Потім селектор 402 може відповідати за вибір режиму за допомогою перемикача 400 режимів для виконання адаптації контексту на основі попередньо одержаних символів або синтаксичних елементів, які відносяться до більшої кількості сусідніх блоків у випадку високоефективного (HE) режиму, порівняно з режимом низької складності (LC), таким чином зменшуючи зберігання надмірної інформації, як тільки що описано.

Далі, описується кодування з меншою складністю різниць векторів руху у відповідності з варіантом виконання.

У стандартному відеокодеці H.264/AVC вектор руху, зв'язаний з макроблоком, передається шляхом сигналізації різниці (різниця векторів руху - mvd) між вектором руху поточного макроблоку і предиктором серединного вектору руху. Коли CABAC використовується як ентропійний кодер, то mvd кодується наступним чином. Ціле значення mvd поділяється на абсолютну величину і знак числа. Абсолютну величину бінаризують з використанням комбінації зрізаного унарного коду і експоненціального коду Голомба 3-го порядку, названих префіксом і суфіксом одержуваного рядка двійкових кодів. Інформаційні величини, які належать до бінаризації зі зрізанням унарного коду, кодуються з використанням моделей контексту, тоді як інформаційні величини, які належать до бінаризації з використанням експоненціальних кодів Голомба, кодуються в байпасному режимі, тобто, з фіксованою ймовірністю 0,5 за допомогою CABAC. Бінаризація з використанням унарних кодів виконується наступним чином. Покладають абсолютну величину цілого значення mvd рівною n потім одержуваний рядок двійкових кодів складається з n '1' і одним залишковим '0'. Як приклад, нехай $n=4$, потім рядком двійкових чисел є '11110'. У випадку зрізаного унарного коду, існує межа і якщо значення перевищує цю межу, то

рядок двійкових чисел складається з $n+1$ '1'. Для випадку mvd , межа дорівнює 9. Це означає, що, якщо кодується абсолютна величина mvd , яка дорівнює або більша 9, то завдяки бінаризації з використанням експоненціальних кодів Голомба одержується 9 '1', рядок двійкових чисел складається з префіксу і суфіксу. Моделювання контексту для зрізаної унарної частини виконується наступним чином. Для першого двійкового числа рядка двійкових чисел беруться абсолютні значення mvd з верхнього і лівого сусіднього макроблоків, якщо вони доступні (якщо не доступні, то розуміється, що значення дорівнює 0). Якщо сума для спеціального компонента (горизонтальний або вертикальний напрям) більша за 2, то вибирається друга модель контексту, якщо абсолютне значення суми більше за 32, то вибирається третя модель контексту, інакше (абсолютне значення суми менше ніж 3) і вибирається перша модель контексту. Окрім того, моделі контексту є різними для кожного компонента. Для другого двійкового числа рядка двійкових чисел використовується четверта модель контексту і п'ята модель контексту використовується для решти двійкових чисел унарної частини. Коли абсолютне значення mvd дорівнює або більша за 9, наприклад, усі двійкові числа зрізаної унарної частини дорівнюють '1', то різниця між абсолютним значенням mvd і 9 кодується в байпасному режимі за допомогою експоненціальних кодів Голомба 3-го порядку. На останньому етапі знак mvd кодується в байпасному режимі.

Найостанніша технологія кодування для mvd при використанні САВАС як ентропійного кодера описується в поточній Тестувальній Моделі (НМ) проекту Високоєфективного кодування відеоданих (HEVC). В HEVC розміри блоків є змінними і форма, специфікована вектором руху, називається прогнозувальним засобом (PU). Розміри PU верхнього лівого сусіднього блока можуть бути іншими, ніж у поточному PU. Тому, як би це не було релевантним, верхній і лівий сусідній блоки тепер називаються верхнім і лівим сусіднім блоком верхнього лівого кута поточного PU. Для самого кодування тільки процес одержання першого двійкового числа може змінюватися у відповідності з варіантом виконання. Замість оцінки абсолютного значення суми MV від сусідніх блоків, кожен сусідній блок може оцінюватися окремо. Якщо абсолютне значення MV сусіднього блока доступне і більше за 16, то індекс моделі контексту може збільшуватися, надаючи ту ж кількість моделей контексту для першого двійкового числа, тоді як кодування решти рівня абсолютного значення MVD і знака є точно таким же що й у стандарті H.264/AVC.

У вищеописаній технології кодування mvd до 9 інформаційних величин повинні кодуватися моделлю контексту, тоді як залишкова величина mvd може кодуватися в байпасному режимі низької складності разом з інформацією про знак. Цей представлений варіант виконання описує технологію зменшення кількості інформаційних величин, закодованих моделями контексту, надаючи більшу кількість байпасних режимів і зменшує кількість моделей контексту, необхідну для кодування mvd . Для цього, критична величина зменшується від 9 до 1 або 2. Це означає тільки, що перша інформаційна величина, яка уточнює чи абсолютне значення mvd більше за нуль, кодується з використанням моделі контексту, а друга інформаційна величина, яка уточнює чи абсолютне значення mvd більше за нуль, кодується першою і друга інформаційна величина, яка уточнює чи абсолютна величина mvd більша за нуль і одиницю, кодується з використанням моделі контексту, тоді як залишкова величина кодується в байпасному режимі і/або з використанням коду VLC. Усі двійкові числа, які одержуються з процесу бінаризації з використанням коду VLC, не використовуючи унарний або усічений унарний код, кодуються з використанням байпасного режиму низької складності. У випадку PIPE, можлива пряма вставка в і з потоку даних. Більше того, якщо подібне є, то може використовуватися інше визначення верхнього і лівого сусіднього блока для одержання кращого вибору моделі контексту для першої інформаційної величини.

В переважному варіанті виконання експоненціальні коди Голомба використовуються для бінаризації решти абсолютних значень MVD компонентів. Для цього, порядок експоненціального коду Голомба змінний. Порядок експоненціального коду Голомба одержується наступним чином. Після одержання і кодування моделі контексту для першої інформаційної величини і, тому, індексу такої моделі контексту, індекс використовується як порядок для бінаризації з використанням експоненціального коду Голомба. У цьому переважному варіанті виконання модель контексту для першої інформаційної величини знаходиться в інтервалі 1-3, надаючи індекс 0-2, який використовуються як порядок експоненціального коду Голомба. Цей переважний варіант виконання може використовуватися для випадку вискоєфективного (HE) режиму.

В альтернативі до вищеописаної технології використання двох раз п'яти контекстів в кодуванні абсолютного значення MVD для кодування 9 двійкових чисел бінаризації з використанням унарного коду, могли б також використовуватися 14 моделей контексту (7 для

кожного компонента). Наприклад, хоча перша і друга інформаційні величини унарної частини могли б кодуватися чотирма різними контекстами, як описано вище, п'ятий контекст міг би використовуватися для третьої інформаційної величини, а шостий контекст міг би використовуватися відносно до четвертої інформаційної величини, тоді як п'ята-дев'ята інформаційні величини кодуються з використанням сьомого контексту. Таким чином, у цьому випадку повинні вимагатися навіть 14 контекстів і просто залишкова величина може кодуватися в байпасному режимі низької складності. Технологія для зменшення кількості інформаційних величин, закодованих моделями контексту, надає більшу кількість байпасних інформаційних величин і зменшує кількість моделей контексту, необхідну для кодування MVD, повинна зменшувати критичну величину, як, наприклад, від 9 до 1 або 2. Це означає, що тільки перша інформаційна величина, яка уточнює чи абсолютне значення MVD більше за нуль, повинен кодуватися з використанням моделі контексту, або чи перша і друга інформаційні величини, які уточнюють чи абсолютне значення MVD більше за нуль і одиницю, повинен кодуватися з використанням відповідної моделі контексту, тоді як залишкова величина кодується кодом VLC. Усі інформаційні величини, які одержуються з процесу бінаризації з використанням коду VLC, кодуються з використанням байпасного режиму низької складності. У випадку PIPE можливе пряме вставляння в і видалення з потоку даних. Окрім того, представлений варіант виконання використовує інше визначення верхнього і лівого сусіднього блока для одержання кращого вибору моделі контексту для першої інформаційної величини. На додаток до цього, моделювання контексту модифікується у такий спосіб, що кількість моделей контексту, необхідна для першої або першої і другої інформаційних величин, зменшується, приводячи до подальшого зменшення пам'яті. Також, оцінювання сусідніх блоків, таких як вищезгаданий сусідній блок, може усуватися, приводячи до збереження лінійного буфера/пам'яті, необхідної для зберігання значень mvd сусідніх блоків. Нарешті, порядок кодування компонентів може поділятися у спосіб, який дозволяє кодування інформаційних величин з префіксом для обох компонентів (тобто, елементів вибірок, кодованих моделями контексту), за яким слідує кодування байпасних інформаційних величин.

В переважному варіанті виконання експоненціальні коди Голомба використовуються для бінаризації решти компонентів абсолютного значення mvd. Для цього порядок експоненціального коду Голомба є змінним. Порядок експоненціального коду Голомба може одержуватися наступним чином. Після одержання моделі контексту для першої інформаційної величини і, тому, після одержання індексу такої моделі контексту, індекс використовується як порядок для бінаризації з використанням експоненціального коду Голомба. У цьому переважному варіанті виконання модель контексту для першої інформаційної величини становить 1-3, надаючи індекс 0-2, який використовується як порядок експоненціального коду Голомба. Цей переважний варіант виконання може використовуватися для випадку вискоєфективного режиму (HE) і кількість моделей контексту зменшується до 6. Для зниження кількості моделей контексту знову і, тому, для збереження пам'яті, горизонтальні і вертикальні компоненти можуть використовувати однакові моделі контексту в подальшому переважному варіанті виконання. У такому випадку вимагається тільки 3 моделі контексту. Окрім того, у подальшому переважному варіанті виконання винаходу для оцінки може братися до уваги тільки лівий сусідній блок.

У цьому переважному варіанті виконання порогова величина може не змінюватися (наприклад, тільки єдина порогова величина 16 надає параметр 0 або 1 експоненціального коду Голомба, або єдина порогова величина 32 надає параметр 0 або 2 експоненціального коду Голомба). Цей переважний варіант виконання зберігає лінійний буфер, необхідний для зберігання значення mvd. В іншому переважному варіанті виконання порогова величина модифікується і дорівнює 2 і 16. Для такого переважного варіанта виконання загалом 3 моделі контексту вимагаються для кодування mvd і можливий параметр експоненціального коду Голомба становить 0-2. В подальшому переважному варіанті виконання порогова величина дорівнює 16 і 32. Знову, описаний варіант виконання придатний для випадку вискоєфективного (HE) режиму.

В подальшому переважному варіанті виконання винаходу критична величина зменшується від 9 до 2. У цьому переважному варіанті виконання перша інформаційна величина і друга інформаційна величина можуть кодуватися з використанням моделей контексту. Вибір моделі контексту для першої інформаційної величини може робитися як і в рівні техніки або модифікуватися у спосіб, описаний у вищенаведеному переважному варіанті виконання. Для другої інформаційної величини окрема модель контексту вибирається як і в рівні техніки. В подальшому переважному варіанті виконання модель контексту для другої інформаційної величини вибирається оцінюванням mvd лівого сусіднього блока. Для такого випадку, індекс

моделі контексту є тим же, що й для першої інформаційної величини, тоді як доступні моделі контексту відмінні від тих, що використовуються для першої інформаційної величини. Загалом вимагається 6 моделей контексту (відзначається, що компоненти використовують спільні моделі контексту). Знову, параметр експоненціального коду Голомба може залежати від вибраного

5 індексу моделі контексту першої інформаційної величини. В іншому переважному варіанті виконання винаходу параметр експоненціального коду Голомба залежить від індексу моделі контексту другої інформаційної величини. Описані варіанти виконання винаходу можуть використовуватися для випадку вискоєфективного (HE) режиму.

В подальшому переважному варіанті виконання винаходу моделі контексту для обох

10 інформаційних величин є фіксованими і не одержуються шляхом оцінки або лівого або вищезгаданих блоків. Для цього переважного варіанта виконання загальна кількість моделей контексту дорівнює 2. В подальшому переважному варіанті виконання винаходу перша інформаційна величина і друга інформаційна величина використовують однакову модель контексту. В результаті, тільки одна модель контексту вимагається для кодування mvd. В обох

15 переважних варіантах виконання винаходу параметр експоненціального коду Голомба може бути фіксованим і рівним 1. Описаний переважний варіант виконання винаходу придатний для обох конфігурацій HE і LC.

В іншому переважному варіанті виконання порядок експоненціального коду Голомба одержується незалежно від індексу моделі контексту першої інформаційної величини. У цьому

20 випадку абсолютне значення суми звичайного вибору моделі контексту стандарту H.264/AVC використовується для одержання порядку для експоненціального коду Голомба. Цей переважний варіант виконання може використовуватися для випадку вискоєфективного режиму (HE).

В подальшому переважному варіанті виконання порядок експоненціальних кодів Голомба є

25 фіксованим і встановлюється рівним 0. В іншому переважному варіанті виконання порядок експоненціальних кодів Голомба є фіксованим і встановлюється рівним 1. В переважному варіанті виконання порядок експоненціальних кодів Голомба фіксується рівним 2. В подальшому варіанті виконання порядок експоненціальних кодів Голомба фіксується рівним 3. В подальшому переважному варіанті виконання порядок експоненціальних кодів Голомба

30 фіксується згідно з формою і розміром поточного PU. Представлені переважні варіанти виконання можуть використовуватися для випадку режиму низької складності (LC). Відзначається, що фіксований порядок експоненціальних кодів Голомба розглядається з меншою кількістю інформаційних величин з моделями контексту.

В переважному варіанті виконання сусідні блоки визначаються наступним чином. Для

35 вищезгаданого PU усі PU, які охоплюють поточний PU, беруться до уваги і використовується PU з найбільшим MV. Це робиться також для лівого сусіднього блока. Усі PU, які охоплюють поточний PU, оцінюються і використовується PU з найбільшим MV. В іншому переважному варіанті виконання середня абсолютна величина вектору руху з усіх PU, які охоплюють верхню і

40 ліву границі, поточний PU використовується для одержання першої інформаційної величини. Для представлених вище переважних варіантів виконання можна змінити порядок кодування наступним чином, mvd повинно специфікуватися для горизонтального і вертикального напрямку один за іншим (або навпаки). Таким чином, потрібно кодувати два рядки інформаційних величин. Для мінімізації кількості перемикачів режимів для ентропійного кодувального засобу (тобто, перемикачів між байпасним і регулярним режимом), можна кодувати інформаційні

45 величини, кодовані моделями контексту для обох компонентів на першому етапі, за яким слідує дві інформаційні величини, кодовані в байпасному режимі на другому етапі. Відзначається, що це тільки переупорядкування.

Будь-ласка, візьміть до уваги, що двійкові числа, які одержуються з бінаризації, яка використовує унарний код або зрізаний унарний код, можуть також представлятися

50 еквівалентними даними бінаризації фіксованої довжини одного ідентифікатора на індекс інформаційної величини, який уточнює, чи величина більша за поточний індекс інформаційної величини. Як приклад, критична величина для бінаризації mvd з використанням зрізаного унарного коду встановлюється рівною 2, що надає кодові слова 0, 10, 11 для величин 0, 1, 2. У відповідному процесі бінаризації інформаційних величин фіксованої довжини з одним

55 ідентифікатором на індекс інформаційної величини, один ідентифікатор для індексу 0 інформаційної величини (тобто, перша інформаційна величина) уточнює чи абсолютне значення mvd більше за 0, чи ні, і один ідентифікатор для другої інформаційної величини з індексом 1 уточнює чи абсолютне значення mvd більше за 1, чи ні. Коли другий ідентифікатор кодується тільки, коли перший ідентифікатор дорівнює 1, то це надає ті ж кодові слова 0, 10, 11.

Далі описується представлення з масштабованою складністю внутрішнього стану моделей ймовірності у відповідності з описаним варіантом виконання.

У HE-PIPE засобі внутрішній стан моделі ймовірності оновлюється після кодування нею інформаційної величини. Оновлений стан одержується таблицею пошуку переходу станів, яка використовує старий стан і величину кодуваної інформаційної величини. У випадку CABAC модель ймовірності може займати 63 різні стани, де кожен стан відповідає моделі ймовірності в інтервалі (0,0, 0,5). Кожен з цих станів використовується для реалізації двох моделей ймовірностей. На додаток до ймовірності, присвоєної стану, 1,0 мінус ймовірність також використовується і ідентифікатор, названий *valMps*, зберігає інформацію про те, чи використовується ймовірність чи 1,0 мінус ймовірність. Це приводить до загалом 126 станів. Для використання такої моделі ймовірності з концепцією PIPE кодування, кожен з 126 станів необхідно перетворювати на один з доступних PIPE кодерів. В поточних втіленнях PIPE кодерів це робиться шляхом використання таблиці пошуку. Приклад такого перетворення зображений в Таблиці А.

Далі описується варіант виконання як внутрішній стан моделі ймовірності може представлятися для уникнення використання таблиці пошуку для перетворення внутрішнього стану на PIPE індекс. Виключно деякі прості операції по маскуванню бітів необхідні для одержання PIPE індексу із змінної внутрішнього стану моделі ймовірності. Це нове представлення з масштабованою складністю внутрішнього стану моделі ймовірності виконане дворівневим. Для застосувань, де операція низької складності є обов'язковою, використовується тільки перший рівень. Вона описує тільки *pipe* індекс і ідентифікатор *valMps*, який використовується для кодування або декодування відповідних інформаційних величин. У випадку описаної схеми PIPE ентропійного кодування перший рівень може використовуватися для диференціації між 8 різними моделями ймовірності. Таким чином, перший рівень повинен потребувати 3 біт для *pipeIdx* і один додатковий біт для ідентифікатора *valMps*. Завдяки другому рівню кожен з великих інтервалів ймовірності першого рівня ділиться на декілька менших інтервалів, які зберігають представлення ймовірностей при вищих роздільних здатностях. Це детальніше представлення дозволяє точнішу роботу оцінювачів ймовірності. Загалом, для кодування вигідно мати високі RD-характеристики. Як приклад, це представлення з масштабованою складністю внутрішнього стану моделей ймовірності з використанням PIPE зображується наступним чином:

Перший Рівень				Другий Рівень			
b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
MPS		PIPE Індекс (0-7)		Індекс деталізації(0-15)			

Перший і другий рівні зберігаються в єдиній 8 бітовій пам'яті. 4 біти використовуються для зберігання першого рівня - індекса, який визначає PIPE індекс з величиною MPS на найбільш значущому біті- а інші 4 біти використовуються для зберігання другого рівня. Для втілення поведінки оцінювача ймовірності CABAC, кожен PIPE індекс має конкретне число дозволених індексів деталізації, яке залежить від того, скільки станів CABAC було перетворено на PIPE індекс. Наприклад, для перетворення в Таблиці А кількість станів CABAC на PIPE індекс вказана в Таблиці В.

PIPE індекс	0	1	2	3	4	5	6	7
Кількість станів CABAC	3	7	5	7	10	14	16	1

Таблиця В: Кількість станів CABAC на PIPE індекс для прикладу Таблиці А.

Під час процесу кодування або декодування інформаційної величини до PIPE індекса і *valMps* можна одержувати прямий доступ за допомогою використання простих операцій маскування бітів або зсуву бітів. Процеси кодування низької складності вимагають тільки 4 біти першого рівня, а процеси вискоефективного кодування можуть додатково використовувати 4 біти другого рівня для реалізації оновлення моделі ймовірності оцінювача ймовірності CABAC. Для виконання цього оновлення, може формуватися таблиця пошуку переходу стану, яка має ті ж переходи станів що й первинна таблиця, але яка використовує дворівневе представлення станів з масштабованою складністю. Первинна таблиця переходу станів складається з двох рядків з 63 елементами в кожному. Для кожного вхідного стану вона містить два вихідні стани. При використанні представлення з масштабованою складністю розмір таблиці переходу станів не перевищує два рядки з 128 елементами в кожному, що є прийнятним збільшенням розміру таблиці. Це збільшення залежить від того скільки бітів використовується для представлення

індексу деталізації і для точної емуляції поведінки оцінювача ймовірності SABAC, при цьому вимагається чотири біти. Однак, міг би використовуватися інший оцінювач ймовірності, який може працювати на меншій множині станів SABAC так, що для кожного ріре індексу допускається не більше ніж 8 станів. Тому, споживання пам'яті може узгоджуватися із заданим рівнем складності процесу кодування шляхом адаптації кількості бітів, використовуваних для представлення індексу деталізації. Порівняно з внутрішнім станом моделей ймовірності з SABAC, де існує 64 індекси стану ймовірності, використання таблиць пошуку для перетворення моделей ймовірності на спеціальний PIPE код усувається і не вимагається додаткове перетворення.

Далі описується оновлення моделі контексту з масштабованою складністю у відповідності з варіантом виконання.

Для оновлення моделі контексту, її індекс стану ймовірності може оновлюватися на основі одного або більшої кількості попередньо кодованих інформаційних величин. При встановленні HE-PIPE це оновлення виконується після кодування або декодування кожної інформаційної величини. І, навпаки, при встановленні LC-PIPE це оновлення може ніколи не виконуватися.

Однак, можна виконувати оновлення моделей контексту з масштабованою складністю. Тобто, рішення, чи оновлювати модель контексту, чи ні, може базуватися на різних аспектах. Наприклад, кодер може не оновлюватися для окремих моделей контексту тільки як, наприклад, для моделей контексту синтаксичного елемента `coeff_significant_flag`, і завжди оновлюється для усіх інших моделей контексту.

Іншими словами, селектор 402 міг би конфігуруватися для символів кожного з ряду наперед визначених типів символів для виконання вибору серед ентропійних декодерів 322 в залежності від відповідної моделі ймовірності, зв'язаної з відповідним наперед визначеним символом так, що кількість наперед визначених типів символів менша в режимі низької складності, ніж у високоефективному режимі.

Окрім того, критеріями для визначення того, чи оновлювати модель контексту, чи ні, могли б бути, наприклад, розмір пакета потоку бітів, кількість інформаційних величин, таким чином декодованих, або чи оновлення виконується тільки після кодування конкретної фіксованої або змінної кількості інформаційних величин для моделі контексту.

За допомогою цієї схеми для прийняття рішення, чи оновлювати моделі контексту, чи ні, може втілюватися оновлення моделі контексту з масштабованою складністю. Це дозволяє збільшення або зменшення частини інформаційних величин в потоці бітів, для яких виконується оновлення моделі контексту. Чим більше оновлень моделі контексту, тим краща ефективність кодування і вища складність обрахунку. Таким чином, описаною схемою може досягатися оновлення моделі контексту з масштабованою складністю.

В переважному варіанті виконання оновлення моделі контексту виконується для інформаційних величин усіх синтаксичних елементів за виключенням синтаксичних елементів `coeff_significant_flag`, `coeff_abs_greater1` і `coeff_abs_greater2`.

В подальшому переважному варіанті виконання оновлення моделі контексту виконується тільки для інформаційних величин синтаксичних елементів `coeff_significant_flag`, `coeff_abs_greater1` і `coeff_abs_greater2`.

В подальшому переважному варіанті виконання оновлення моделі контексту виконується для усіх моделей контексту при початку кодування або декодування частини масиву даних. Після обробки конкретної наперед визначеної кількості блоків перетворення, оновлення моделі контексту деактивується для усіх моделей контексту до досягання кінця частини масиву даних.

Наприклад, селектор 402 може конфігуруватися для символів наперед визначеного типу для виконання вибору серед ентропійних декодерів 322 в залежності від моделі ймовірності, зв'язаної з наперед визначеним типом символу, разом з або без оновлення відповідної моделі ймовірності так, що довжина фази вивчення послідовності символів, на якій виконується вибір для символів наперед визначеного типу разом з оновленням, менша в режимі низької складності ніж у високоефективному режимі.

Подальший переважний варіант виконання ідентичний попередньому описаному переважному варіанту виконання, але він використовує представлення внутрішнього стану моделей контексту з масштабованою складністю у такий спосіб, що одна таблиця зберігає "першу частину" (`valMps` і `ripeldx`) усіх моделей контексту, а друга таблиця зберігає "другу частину" (`refineldx`) усіх моделей контексту. В місці, де оновлення моделі контексту деактивується для усіх моделей контексту (як описано в попередньому переважному варіанті виконання), більше не потрібне зберігання таблицею "другої частини" і воно може не братися до уваги.

Далі описується оновлення моделі контексту для послідовності інформаційних величин у відповідності з варіантом виконання.

В конфігурації LC-PIPE інформаційні величини синтаксичних елементів типу `coeff_significant_flag`, `coeff_abs_greater1` і `coeff_abs_greater2` групуються в підмножини. Для кожної підмножини використовується єдина модель контексту для кодування її інформаційних величин. У цьому випадку, оновлення моделі контексту може виконуватися після кодування

5

фіксованої кількості інформаційних величин цієї послідовності. Це оновлення моделі з багатьма інформаційними величинами описується далі. Однак, це оновлення може відрізнитися від оновлення, яке використовує тільки останню кодовану інформаційну величину і внутрішній стан моделі контексту. Наприклад, для кожної інформаційної величини, яка кодувалася, виконується один етап оновлення моделі контексту.

10

`b b b b b b b b`

У випадку HE-PIPE після декодування кожної інформаційної величини виконується оновлення моделі контексту:

15

`b u b u b u b u b u b u b u`

Для того, щоб дещо знизити складність, оновлення моделі контексту може виконуватися після послідовності інформаційних величин (у цьому прикладі після кожних 4 інформаційних величин виконуються оновлення цих 4 інформаційних величин):

20

`b b b b u u u u b b b b u u u u`

Тобто, селектор 402 може конфігуруватися для символів наперед визначеного типу для виконання вибору серед ентропійних декодерів 322 в залежності від моделі ймовірності, зв'язаної з наперед визначеним типом символу, разом з або без оновлення відповідної моделі ймовірності так, що частота, з якою виконується вибір для символів наперед визначеного типу разом з оновленням, нижча в режимі низької складності, ніж у високоефективному режимі.

25

У цьому випадку після декодування 4 інформаційних величин слідує 4 етапи оновлення на основі 4 тільки що декодованих інформаційних величин. Відзначається, що ці чотири етапи оновлення можуть виконуватися за один єдиний етап шляхом використання спеціальної таблиці пошуку. Ця таблиця пошуку зберігає для кожної можливої комбінації з 4 інформаційних величин і кожного можливого внутрішнього стану моделі контексту одержуваний новий стан після чотирьох традиційних етапів оновлення.

30

В певному режимі оновлення з багатьма інформаційними величинами використовується для синтаксичного елемента `coeff_significant_flag`. Для інформаційних величин усіх інших синтаксичних елементів не використовується оновлення моделі контексту. Кількість інформаційних величин, які кодуються перед виконанням етапу оновлення з багатьма елементами вибірки, встановлюється рівною p . Коли кількість інформаційних величин множини не ділиться на p , то 1 - $p-1$ інформаційних величин залишаються в кінці підмножини після останнього оновлення з багатьма елементами вибірки. Для кожної з цих інформаційних величин традиційне оновлення з однією інформаційною величиною виконується після кодування усіх цих інформаційних величин. Кількість p може бути будь-яким додатнім числом, більшим за 1. Інший режим може бути ідентичним з попереднім режимом за виключенням того, що оновлення з багатьма інформаційними величинами виконується для довільних комбінацій `coeff_significant_flag`, `coeff_abs_greater1` і `coeff_abs_greater2` (тільки замість `coeff_significant_flag`).

35

40

45

Таким чином, цей режим повинен бути складнішим, ніж інший. Усі інші синтаксичні елементи (де не використовується оновлення з багатьма інформаційними величинами) можуть ділитися на дві окремі підмножини, де для однієї з підмножин використовується оновлення з єдиною інформаційною величиною, а для іншої підмножини не використовується оновлення моделі контексту. Будь-які можливі окремі підмножини є дійсними (включаючи порожню підмножину).

50

В альтернативному варіанті виконання оновлення з багатьма інформаційними величинами може базуватися тільки на останніх m інформаційних величинах, які кодуються безпосередньо перед етапом оновлення з багатьма інформаційними величинами, m може бути будь-яким натуральним числом, меншим за p . Таким чином, декодування може виконуватися наступним чином:

b b b b u u b b b b u u b b b b u u b b b b...

з $n=4$ і $m=2$.

Тобто, селектор 402 може конфігуруватися для символів наперед визначеного типу для виконання вибору серед ентропійних декодерів 322 в залежності від моделі ймовірності, зв'язаної з наперед визначеним типом символу, разом з оновленням відповідної моделі ймовірності, при цьому кожен n -й символ наперед визначеного типу базується на m найостанніших символів наперед визначеного типу так, що відношення n/m більше в режимі низької складності, ніж у високоефективному режимі.

В подальшому переважному варіанті виконання для синтаксичного елемента coeff_significant_flag схема моделювання контексту, яка використовує локальний зразок, як описано вище для конфігурації HE-PIPE, може використовуватися для присвоєння моделей контексту інформаційним величинам синтаксичного елемента. Однак, для цих інформаційних величин не використовується оновлення моделі контексту.

Окрім того, селектор 402 може конфігуруватися для символів наперед визначеного типу для вибору одного з ряду контекстів в залежності від кількості попередньо одержаних символів послідовності символів і для виконання вибору серед ентропійних декодерів 322 в залежності від моделі ймовірності, зв'язаної з вибраним контекстом, так, що кількість контекстів і/або кількість попередньо одержаних символів менша в режимі низької складності, ніж у високоефективному режимі.

Ініціалізація моделі ймовірності, яка використовує 8 бітові ініціалізуючі величини

Цей розділ описує процес ініціалізації внутрішнього стану моделей ймовірності з масштабованою складністю, використовуючи так звану 8 бітову ініціалізуючу величину замість двох 8 бітових величин як у випадку стандарту кодування відеоданих рівня техніки H.265/AVC. Він складається з двох частин, які співставні з парами ініціалізуючих величин, використовуваними для моделей ймовірності в CABAC стандарту H.264/AVC. Дві частини представляють два параметри лінійного рівняння для обрахунку початкового стану моделі ймовірності, представляючи конкретну ймовірність (наприклад, у формі PIPE індексу) з QP:

- Перша частина описує крутизну і вона використовує залежність внутрішнього стану від параметра дискретизації (QP), який використовується під час кодування або декодування.

- Друга частина визначає PIPE індекс при заданому QP, а також valMps.

Два різні режими доступні для ініціалізації моделі ймовірності з використанням заданої ініціалізуючої величини. Перший режим є незалежною від QP ініціалізацією. Він використовує тільки PIPE індекс і valMps, визначені у другій частині ініціалізуючої величини для усіх QP. Це ідентично випадку, де крутизна дорівнює 0. Другий режим є залежною від QP ініціалізацією і він додатково використовує крутизну першої частини ініціалізуючої величини для зміни PIPE індексу і для визначення індексу деталізації. Дві частини 8 бітової ініціалізуючої величини зображені далі:

Перша частина				Друга частина			
b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
Індекс		Крутизни		PIPE Індекс Ймовірності			

Він складається з двох 4 бітових частин. Перша частина містить індекс, який вказує 1 з 16 різних наперед визначених величин крутизни, які зберігаються в масиві. Наперед визначені величини крутизни складаються з 7 від'ємних величин крутизни (індекс крутизни 0-6): одна величина крутизни, яка дорівнює нулю (індекс крутизни 7), і 8 додатних величин крутизни (індекс крутизни 8-15). Величини крутизни вказані в Таблиці С.

Таблиця

Індекс Крутизни	0	1	2	3	4	5	6	7
Величина Крутизни	-239	-143	-85	-51	-31	-19	-11	0
Індекс Крутизни	8	9	10	11	12	13	14	15
Величина Крутизни	11	19	31	51	85	143	239	399

- Усі величини масштабовані коефіцієнтом 256 для уникнення використання операцій з плаваючою точкою. Друга частина є PIPE індексом, який втілює висхідну ймовірність $valMps=1$ між інтервалом ймовірності $p=0$ і $p=1$. Іншими словами, PIPE кодер n повинен працювати при вищій моделі ймовірності ніж PIPE кодер $n-1$. Для кожної моделі ймовірності доступний один PIPE індекс ймовірності і він ідентифікує PIPE кодер, чий інтервал ймовірності містить ймовірність $P_{valMPS}=1$ для $QP=26$.

Таблиця D:

Перетворення другої частини ініціалізуючої величини на PIPE кодери і $valMps$: UR=перетворення унарного коду на код Райса, TB=потрійний двійковий код, BP=двійковий ріре-код, EP= однакова ймовірність (некодована)

PIPE Ймовірності	Індекс	0	1	2	3	4	5	6	7
PIPE Кодер		UR5	UR4	UR3	UR2	TB	BP2	BP3	EP
MPS		0	0	0	0	0	0	0	0
PIPE Ймовірності	Індекс	8	9	10	11	12	13	14	15
PIPE Кодер		EP	BP3	BP2	TB	UR2	UR3	UR4	UR5
MPS		1	1	1	1	1	1	1	1

- QP і 8 бітова ініціалізуюча величина необхідні для обрахунку параметрів ініціалізації внутрішнього стану моделей ймовірності шляхом вирішення простого лінійного рівняння у формі $y=m*(QP-QPref)+256*b$. Відзначається, що m визначає крутизну, яка береться з Таблиці С шляхом використання індексу крутизни (перша частина 8 бітової ініціалізуючої величини) і b позначає PIPE кодер при $QPref=26$ (друга частина 8 бітової ініціалізуючої величини: "PIPE Індекс Ймовірності"). Потім, $valMPS$ дорівнює 1 і $pipeldx$ дорівнює $(y-2048) >> 8$, якщо y більший за 2047. Інакше, $valMPS$ дорівнює 0 і $pipeldx$ дорівнює $(2047-y) >> 8$. Індекс деталізації дорівнює $((y-2048) \& 255) * numStates >> 8$, якщо $valMPS$ дорівнює 1. Інакше, індекс деталізації дорівнює $((2047-y) \& 255) * numStates >> 8$. В обох випадках, $numStates$ дорівнює кількості станів CABAC ріре індекс, як вказано в Таблиці B.
- Вищенаведена схема може не тільки використовуватися в комбінації з PIPE кодерами, але й також в зв'язку з вищезгаданими схемами CABAC. За відсутності PIPE кількість станів CABAC, тобто, станів ймовірності, між якими реалізується перехідний стан в оновленні ймовірності ($pState_current[bin]$) на PIPE індекс (тобто, відповідні найбільш значущі біти $pState_current[bin]$) припадає потім тільки множина параметрів, які, фактично, реалізують кусково-лінійну інтерполяцію стану CABAC в залежності від QP. Окрім того, ця кусково-лінійна інтерполяція може також практично усуватися у випадку, де параметр $numStates$ використовує однакову величину для усіх PIPE індексів. Наприклад, встановлення $numStates$ рівним 8 для усіх випадків надає загалом $16*8$ станів і обрахунок індексу деталізації спрощується до $((y-2048) \& 255) >> 5$ для $valMPS$, рівного 1, або $((2047-y) \& 255) >> 5$ для $valMPS$, рівного 0. Для цього випадку перетворення представлення з використанням $valMPS$, PIPE індексу та індексу деталізації назад на представлення, використовуване оригінальним CABAC стандарту H.264/AVC, є дуже простим. Стан CABAC заданий як (PIPE індекс $<< 3$) + індекс деталізації. Цей аспект описується далі нижче по відношенню до Фіг. 16.
- Не дивлячись на те, що крутизна 8 бітової ініціалізуючої величини дорівнює нулю, або не дивлячись на те, що QP дорівнює 26, необхідно обрахувати внутрішній стан шляхом використання лінійного рівняння з QP процесу кодування або декодування. У випадку крутизни, яка дорівнює нулю, або такого QP поточного процесу кодування, що дорівнює 26, друга частина 8 бітової ініціалізуючої величини може безпосередньо використовуватися для ініціалізації внутрішнього стану моделі ймовірності. Інакше, десяткова частина одержуваного внутрішнього стану може потім використовуватися для визначення індексу деталізації у вискоефективних кодуваннях за допомогою лінійної інтерполяції між межами спеціального PIPE кодера. У цьому переважному варіанті виконання лінійну інтерполяцію здійснюють простим множенням десяткової частини на загальну кількість індексів деталізації, доступних для поточного PIPE кодера і перетворенням результату на найближчий цілий індекс деталізації.
- Процес ініціалізації внутрішнього стану моделей ймовірності може змінюватися відносно до кількості PIPE станів індексу ймовірності. Зокрема, появи два рази однакового ймовірного

- режиму, який використовує PIPE кодер E1, тобто, використання двох різних PIPE індексів для розрізнення між MPS, який становить 1 або 0, можна уникати наступним чином. Знову, процес може активуватися під час запуску синтаксичного аналізу частини масиву даних, а вхідні дані цього процесу можуть бути 8 бітовою ініціалізуючою величиною, як вказано в Таблиці Е, яка повинна, наприклад, передаватися в потоці бітів для кожної моделі контексту, яка ініціалізується.

Таблиця Е:

Встановлення 8 бітів Ініціалізуючої Величини для моделі ймовірності

Біти Ініціалізуючої Величини	Перші 4 біти				Останні 4 біти			
	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	b ₀
Змінна	Індекс крутизни				Індекс Ймовірності			

- Перші 4 біти визначають індекс крутизни і одержуються маскуванням бітів b₄-b₇. Для кожного індексу крутизни крутизна (m) уточнюється і відображається в Таблиці F.

Таблиця F

Величини змінної m для Індексу крутизни

Індекс крутизни	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
m	-239	-143	-85	-51	-31	-19	-11	0	11	19	31	51	85	143	239	399

- Біти b₀ -b₃, останні 4 біти 8 бітової ініціалізуючої величини ідентифікують probldx (індекс ймовірності) і описують ймовірність при визначеній QP, probldx 0 вказує найвищу ймовірність для символів з величиною 0 і, відповідно, probldx 14 вказує найвищу ймовірність для символів з величиною 1. Таблиця G показує для кожного probldx відповідний ріре кодер і його valMps.

Таблиця G

Перетворення останньої 4 бітової частини ініціалізуючої величини на PIPE кодери і valMps: UR = перетворенню унарного коду на код Райса, TB = потрібний двійковий код, BP = двійковий ріре код, EP = однакова ймовірність (некодована)

probldx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Pipe кодер	UR5	UR4	UR3	JR2	TBC	BP2	BP3	EP	BP3	BP2	TBC	UR2	UR3	UR4	UR5
valMps	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

- За допомогою обох величин обрахунків внутрішнього стану може виконуватися шляхом використання лінійного рівняння такого як $y = m \cdot x + 256 \cdot b$, де m позначає крутизну, x позначає QP поточної частини масиву даних і b одержується з probldx, як вказано в наступному описі. Усі величини у цьому процесі масштабовані коефіцієнтом 256 для уникнення використання операцій з плаваючою точкою. Вихідні дані (y) цього процесу представляють внутрішній стан моделі ймовірності при поточній QP і зберігаються у 8 бітовій пам'яті. Як зображено в Таблиці G, внутрішній стан складається з valMPs, pipeldx і refineldx.

Таблиця H

Встановлення внутрішнього стану моделі ймовірності

Біти ініціалізуючої величини	Перші 4біти				Останні 4 біти			
	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	b ₀
Змінна	valMps		pipeldx		refineldx			

Присвоєння `refineldx` і `pipeldx` подібне до внутрішнього стану моделей ймовірності CABAC (`pStateCtx`) і представлено в Таблиці І.

Таблиця І

Присвоєння `pipeldx`, `refineldx` і `pStateCtx`

<code>pipeldx</code>	0			1								2					
<code>refineldx</code>	0	1	2	0	1	2	3	4	5	6		0	1	2	3	4	
<code>pStateCtx</code>	0	1	2	3	4	5	6	7	8	9		10	11	12	13	14	
<code>pipeldx</code>	3							4									
<code>refineldx</code>	0	1	2	3	4	5	6	0	1	2	3	4	5	6	7	8	9
<code>pStateCtx</code>	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
<code>pipeldx</code>	5																
<code>refineldx</code>	0	1	2	3	4	5	6	7	8	9	10	11	12	13			
<code>pStateCtx</code>	32	33	34	35	36	37	38	39	40	41	42	43	44	45			
<code>pipeldx</code>	6																7
<code>refineldx</code>	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0
<code>pStateCtx</code>	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62

В переважному варіанті виконання `probdx` визначається при $QP=26$. На основі 8 бітової ініціалізуючої величини внутрішній стан (`valMps`, `pipeldx` і `refineldx`) моделі ймовірності обробляється, як описано в наступному псевдокоді:

```
n=( probdx<<8 )-m*26
fullCtxState=max( 0, min( 3839, ( m*max( 0,min( 51, SliceQPv ) ) ) )+n+128 )
```

```
remCtxState=fullCtxState & 255
```

```
preCtxState=fullCtxState>>8
```

```
if( preCtxState<8) {
```

```
  pipeldx=7-preCtxState
```

```
  valMPS=0
```

```
}else {
```

```
  pipeldx=preCtxState-8
```

```
  valMPS=1
```

```
}
```

```
offset={3, 7, 5, 7, 10, 14, 16, 1 }
```

```
if( pipeldx= 0){
```

```
  if( remCtxState<=127)
```

```
    remCtxState=127-remCtxState
```

```
  else
```

```
    remCtxState=remCtxState - 128
```

```
  refineldx= (remCtxState<<1)*offset>>8
```

```
}else {
```

```
  if( valMPS= 0)
```

```
    remCtxState = 255 - remCtxState
```

```
    refineldx= (remCtxState*offset[pipeldx])>>8
```

```
}
```

Як показано в псевдокоді, `refineldx` обраховується лінійною інтерполяцією між інтервалом `pipeldx` і дискретизацією з наданням відповідного `refineldx`. Зміщення специфікує загальну кількість `refineldx` для кожного `pipeldx`. Інтервал $[7, 8)$ $fullCtxState/256$ ділиться навпіл. Інтервал $[7, 7.5)$ перетворюється на `pipeldx=0` і `valMps=0`, а інтервал $[7.5, 8)$ перетворюється на `pipeldx=0` і `valMps=1`. Фіг. 15 зображає процес одержання внутрішнього стану і відображає перетворення $fullCtxState/256$ на `pStateCtx`.

Відзначається, що крутизна вказує залежність `probdx` і QP . Якщо `slopeldx` 8 бітової ініціалізуючої величини дорівнює 7, то одержуваний внутрішній стан моделі ймовірності є таким же як і для усіх QP (тому, процес ініціалізації внутрішнього стану залежить від поточної QP частини масиву даних).

Тобто, селектор 402 може ініціалізувати `pipe` індекси, які використовуються в декодуванні наступної частини потоку даних, такого як увесь потік або наступна частина масиву даних, використовуючи синтаксичний елемент, який вказує розмір кроку дискретизації QP , використовуюваного для дискретизації даних цієї частини, таких як рівні значущості коефіцієнта

перетворення, які містяться в ньому, використовуючи цей синтаксичний елемент як індекс в таблиці, яка може бути спільною для обох режимів LC і HE. Таблиця, така як таблиця D, може містити ріре індекси для кожного типу символу, для відповідного еталону QPpref або інші дані для кожного типу символу. В залежності від реальної QP поточної частини селектор може

5 обраховувати величину ріре індексу, використовуючи введення чисел а у відповідну таблицю, індексовану фактичною QP і самою QP, таке як множення а на (QP-QPpref). Відмінністю в режимі LC і HE є тільки: селектор обраховує результат просто з нижчою точністю у випадку LC порівняно з режимом HE. Селектор може, наприклад, просто використовувати цілу частину результату обрахунку. В режимі HE залишок вищої точності, такий як дробова частина,

10 використовується для вибору одного з доступних індексів деталізації для відповідного ріре індексу, як вказано нижчою точністю або цілою частиною. Індекс деталізації використовується в режимі HE (потенційно рідше також в режимі LC) для виконання адаптації ймовірності шляхом використання вищезгаданого дрейфу таблиці. При полишенні доступних індексів для поточного ріре індексу на верхній границі, потім вищий ріре індекс вибирають далі з мінімізацією індексу деталізації.

15 При полишенні доступних індексів для поточного ріре індексу на нижній границі, потім наступний нижній ріре індекс вибирають далі з мінімізацією індексу деталізації до максимум доступного для нового ріре індексу. Ріре індекс разом з індексом деталізації визначають стан ймовірності, але для вибору серед часткових потоків, селектор просто використовує ріре індекс. Індекс деталізації просто служить для ближчого або точнішого

20 відслідковування ймовірності.

Однак, вищенаведене обговорення також показало, що масштабована складність може досягатися незалежно від концепції кодування PIPE з Фіг. 7-10 або CABAC, використовуючи декодер, як зображено на Фіг. 12. Декодер з Фіг. 12 передбачений для декодування потоку даних 601, у якому кодуються мультимедійні дані, і містить перемикач режимів 600,

25 сконфігурований для активування режиму низької складності або високоефективного режиму в залежності від потоку даних 601, а також десимволізатор 602, сконфігурований для десимволізації послідовності 603 символів, одержаних або безпосередньо або ентропійним декодуванням, наприклад з потоку даних 601, для одержання цілих синтаксичних елементів 604 з використанням функції перетворення, контрольованої контрольним параметром, для

30 перетворення області слів символної послідовності на кообласть цілих синтаксичних елементів. Реконструктор 605 конфігурується для відновлення мультимедійних даних 606 на основі цілих синтаксичних елементів. Десимволізатор 602 конфігурується для виконання десимволізації так, що контрольний параметр змінюється у відповідності з потоком даних з першою швидкістю у випадку високоефективного режиму, який активується, і контрольний параметр є сталим незалежно від потоку даних або змін, які залежать від потоку даних, але при

35 другій швидкості, нижчій за першу швидкість, у випадку режиму низької складності, який активується, як вказано стрілкою 607. Наприклад, контрольний параметр може змінюватися у відповідності з попередньо десимволізованими символами.

Деякі з вищезгаданих варіантів виконання використовували аспект з Фіг. 12. Синтаксичні елементи `coeff_abs_minus3` і `MVD` в послідовності 327, наприклад, бінаризувалися в десимволізаторі 314 в залежності від режиму, вибраного, як вказано позицією 407, реконструктор 605 використовував ці синтаксичні елементи для відновлення. Очевидно, що обидва аспекти з Фіг. 11 і 19 легко поєднуються, але аспект з Фіг. 12 може також об'єднуватися з іншими кодувальними середовищами.

40

Дивіться, наприклад, вищевказане кодування різниці векторів руху. Десимволізатор 602 може конфігуруватися так, що функція перетворення використовує усічений унарний код для виконання перетворення в першому інтервалі області цілих синтаксичних елементів нижче критичної величини і комбінацію префікса у формі зрізаного унарного коду для критичної величини і суфікса у формі кодового слова VLC в другому інтервалі області цілих синтаксичних

50 елементів включно і з перевищенням критичної величини, при цьому декодер може містити ентропійний декодер 608, сконфігурований для одержання кількості перших інформаційних величин усіченого унарного коду з потоку даних 601 з використанням ентропійного декодування із змінною оцінкою ймовірності і ряду других інформаційних величин кодового слова VLC з використанням байпасного режиму із сталою однаковою ймовірністю. В режимі HE ентропійне кодування може бути складнішим ніж при кодуванні в режимі LC, як вказано стрілкою 609. Тобто, адаптивність до контексту і/або адаптація ймовірності можуть застосовуватися в режимі HE і уникатися в режимі LC або складність може масштабуватися в інших термінах, як визначено вище відносно різних варіантів виконання.

55

Кодек, який підключається до декодера з Фіг. 11, для кодування мультимедійних даних з одержанням потоку даних зображень на Фіг. 13. Він може містити ввідний засіб 500,

60

сконфігурований для сигналізації в потоці даних 501 активування режиму низької складності або високоефективного режиму, конструктор 504, сконфігурований для попереднього кодування мультимедійних даних 505 з одержанням послідовності 506 синтаксичних елементів, символізатор 507, сконфігурований для символізації послідовності 506 синтаксичних елементів з одержанням послідовності 508 символів, певну кількість ентропійних кодерів 310, кожен з яких сконфігурований для перетворення часткових послідовностей символів на кодові слова потоку даних, і селектор 502, сконфігурований для надсилання кожного символу послідовності 508 символів до вибраного одного з певної кількості ентропійних кодерів 310, при цьому селектор 502 конфігурується для виконання вибору в залежності від активованого режиму, вибраного серед режиму низької складності і високоефективного режиму, як вказано стрілкою 511. Перемежувач 510 може необов'язково передбачатися для перемежовування кодових слів кодерів 310.

Кодер, який підключений до декодера з Фіг. 12, для кодування мультимедійних даних з одержанням потоку даних зображений на Фіг. 14 як той, що містить ввідний засіб 700, сконфігурований для сигналізації в потоці даних 701 активування режиму низької складності або високоефективного режиму, конструктор 704, сконфігурований для попереднього кодування мультимедійних даних 705 з одержанням послідовності 706 синтаксичних елементів, які включають цілий синтаксичний елемент, і символізатор 707, сконфігурований для символізації цілого синтаксичного елемента з використанням функції перетворення, контрольованої контрольним параметром, для перетворення області цілих синтаксичних елементів на кообласть слів символічної послідовності, при цьому символізатор 707 конфігурується для виконання символізації так, що контрольний параметр змінюється у відповідності з потоком даних з першою швидкістю у випадку високоефективного режиму, який активується, і контрольний параметр є сталим незалежно від потоку даних або змін в залежності від потоку даних, але з другою швидкістю, нижчою за першу швидкість, у випадку режиму низької складності, який активується, як зображено стрілкою 708. Результат символізації є кодування з одержанням потоку даних 701.

Знову слід нагадати, що варіант виконання з Фіг. 14 легко перетворюється на вищезгаданий варіант виконання адаптивного до контексту двійкового арифметичного кодування/декодування: селектор 509 і ентропійні кодери 310 повинні об'єднуватися в адаптивний до контексту двійковий арифметичний кодер, який повинен безпосередньо видавати потік даних 401 і вибирати контекст для інформаційної величини, який на даний момент одержується з потоку даних. Це особливо справедливо для адаптивності до контексту і/або адаптивності до ймовірності. Обидві функції/адаптивності можуть деактивуватися або розроблятися більш послабленими під час режиму низької складності.

Вище коротко зазначено, що здатність перемикати режими, пояснена відносно до деяких з вищенаведених варіантів виконання, може, у відповідності з альтернативними варіантами виконання, не братися до уваги. Для прояснення цього факту, робиться посилання на Фіг. 16, яка підсумовує вищенаведений опис до такої міри, до якої просте видалення здатності перемикати режими відрізняє варіант виконання з Фіг. 16 від вищеописаних варіантів виконання. Більше того, наступний опис буде розкривати переваги, які одержуються з ініціалізації оцінок ймовірності контекстів з використанням менш точних параметрів для крутизни і зміщення порівняно з, наприклад, стандартом H.264.

Зокрема, Фіг. 16 зображає декодер для декодування відеоданих 405 з потоку даних 401, у який кодуються горизонтальні і вертикальні компоненти різниць векторів руху з використанням бінаризацій горизонтальних і вертикальних компонентів, при цьому бінаризації зрівнюють зрізаний унарний код горизонтальних і, відповідно, вертикальних компонентів в першому інтервалі області горизонтальних і вертикальних компонентів нижче критичної величини, і комбінацію префікса у формі зрізаного унарного коду і критичну величину та суфікс у формі експоненціального коду Голомба горизонтальних і, відповідно, вертикальних компонентів в другому інтервалі області горизонтальних і вертикальних компонентів включно з і вище критичної величини, при цьому критична величина дорівнює 2, а експоненціальний код Голомба має порядок 1. Декодер містить ентропійний декодер 409, сконфігурований для одержання ряду інформаційних величин 326 бінаризацій з потоку даних 401 з використанням двійкового ентропійного декодування шляхом вибору контексту серед різних контекстів і оновлення станів ймовірності, зв'язаних з різними контекстами, залежними від попередньо декодованих частин потоку даних 401. Для більшої точності, як описано вище, ентропійний декодер 409 може конфігуруватися для одержання ряду інформаційних величин 326 бінаризацій з потоку даних 401 з використанням двійкового ентропійного декодування, такого як вищезгадана схема CABAC, або двійкового PIPE декодування, тобто, використовуючи конструкцію, яка

використовує декілька паралельно працюючих ентропійних декодерів 322 разом з відповідним селектором/присвоювачем. Десимволізатор 314 дебіналізує результати бінаризацій синтаксичних елементів різниці векторів руху для одержання цілих величин горизонтальних і вертикальних компонентів різниць векторів руху, а реконструктор 404 відновлює відеодані на основі цілих чисел горизонтальних і вертикальних компонентів різниць векторів руху.

Для детальнішого пояснення цієї процедури, коротке посилання робиться на Фіг. 18. Позиція 800 ілюстративно вказує одну різницю векторів руху, тобто, вектора, який представляє залишок прогнозування між прогнозованим вектором руху і реальним/відновленим вектором руху. Також зображені горизонтальні і вертикальні компоненти 802x і 802y. Вони могли б виражатися в положеннях пікселів, тобто, в кроках пікселя, або положеннях субпікселів, таких як половина кроку пікселя або його четверта частина, або подібне. Горизонтальні і вертикальні компоненти 802x,y є цілими величинами. Їх область проходить від нуля до нескінченності. Знаком можна оперувати окремо і тут він далі не розглядається. Іншими словами, наведений тут опис фокусується на величині різниць векторів руху 802x, y. Область зображена в позиції 804. Справа від осі 804 області Фіг. 19 зображає зв'язані з можливими величинами компонентів 802x,y, розташованими по вертикалі один над іншим, результати бінаризацій, на які перетворюється відповідна можлива величина (біналізується). Як можна побачити, нижче критичної величини, яка дорівнює 2, знаходиться просто усічений унарний код 806, тоді як результат бінаризації має як суфікс також експоненціальний код Голомба 808 з можливих величин, які дорівнюють або більші за критичну величину, яка дорівнює 2, для продовження бінаризації для решти цілої величини, яка перевищує значення, що дорівнює критичній величині мінус 1. Для усіх цих інформаційних величин надається просто два контексти: один для першого положення інформаційної величини результатів бінаризацій горизонтальних і вертикальних компонентів 802x,y, і додатковий один контекст для другого положення інформаційної величини усіченого унарного коду 806 як горизонтальних так і вертикальних компонентів 802x,y. Для положення інформаційної величини експоненціального коду Голомба 808, ентропійним декодером 409 використовується рівномірний байпасний режим. Тобто, припускається, що обидві інформаційні величини мають однакову ймовірність появи. Оцінка ймовірності для цих інформаційних величин є фіксованою. Порівняно з цим, оцінка ймовірності, зв'язана з тільки що згаданими двома контекстами інформаційних величин зрізаного унарного коду 806, неперервно адаптується під час декодування.

Перед детальнішим описом, як ентропійний декодер 409 міг би втілюватися у відповідності з вищенаведеним описом для виконання тільки що згаданих завдань, опис тепер фокусується на можливому втіленні реконструктора 404, який використовує різниці векторів руху 800 і їх цілі величини, у тому вигляді, у якому вони одержуються десимволізатором 314 шляхом повторної бінаризації інформаційних величин кодів 106 і 108, яка зображена на Фіг. 18 з використанням стрілки 810. Зокрема, реконструктор 404 може, як описано вище, одержувати з потоку даних 401 інформацію, яка стосується під розбиття поточно відновленої картини на блоки, серед яких принаймні деякі з них піддаються прогнозуванню з компенсацією руху. Фіг. 19 зображає картинку, яка ілюстративно відновлюється в компоненті 820, і блоки тільки що згаданого підрозбиття картини 120, для якої використовується прогнозування з компенсацією руху для прогнозування контенту картини в блоці 822. Як описано відносно до Фіг. 2A-2C, існують різні можливості для підрозбиття і одержання розмірів блоків 122. Для уникнення передачі для різниці векторів руху 800 для кожного з цих блоків 122, реконструктор 404 може використовувати концепцію злиття, згідно з якою потік даних додатково передає інформацію про злиття на додаток до інформації про підрозбиття або, за відсутності інформації про підрозбиття, на додаток до того факту, що підрозбиття є фіксованим. Інформація про злиття сигналізує реконструктору 404 який з блоків 822 формує групи для злиття. Завдяки цьому заходу реконструктор 404 може застосовувати певну різницю векторів руху 800 до усієї групи блоків 822 для злиття. Зазвичай, на кодувальній стороні передача інформації про злиття знаходить компроміс між надлишком передачі інформації про підрозбиття (якщо такий є), надлишком передачі інформації про злиття і надлишком передачі різниці векторів руху, який послаблюється із збільшенням розміру груп для злиття. З іншого боку, зростаюча кількість блоків на групу для злиття зводить адаптацію різниці векторів руху для цієї групи для злиття до реальних потреб в окремих блоках відповідної групи для злиття, таким чином надаючи менш точні прогнози різниць векторів руху цих блоків з компенсацією руху і потребуючи більшого надлишку передачі інформації для передачі залишку прогнозування у формі, наприклад, рівня значущості коефіцієнта перетворення. Відповідно, на кодувальній стороні належним чином знаходять компроміс. Однак, в будь-якому випадку, концепція злиття надає різниці векторів руху для груп для злиття, які проявляють меншу просторову інтеркореляцію. Дивіться, наприклад,

Фіг. 19, яка показує шляхом затінення приналежність до певної групи для злиття. Очевидно, що реальний рух контенту картинки в цих блоках був таким подібним, що кодувальна сторона вирішила об'єднати відповідні блоки. Однак, кореляція з рухом контенту картинки в інших групах для злиття є низькою. Відповідно, обмеження на використання просто одного контексту на інформаційну величину зрізаного унарного коду 806 негативно не впливає на ефективність ентропійного кодування, оскільки концепція злиття вже в достатній мірі забезпечує просторову інтеркореляцію між рухом сусідніх контентів картинки. Контекст може просто вибиратися на основі того факту, що інформаційна величина є частиною бінаризації компонента 802x, у різниці векторів руху і на основі положення інформаційної величини, яке є або 1 або 2 завдяки критичній величині, яка дорівнює двом. Відповідно, інші вже декодовані інформаційні величини/синтаксичні елементи/mvd компоненти 802x,у не впливають на вибір контексту.

Подібним чином, реконструктор 404 може конфігуруватися для зменшення інформаційного контенту, який передається у формі різниць векторів руху додатково (окрім просторового і/або тимчасового прогнозу векторів руху) шляхом використання концепції багатогіпотезного прогнозування, згідно з якою, перш за все, для кожного блока або групи для злиття генерується список предикторів векторів руху з подальшою явною або неявною передачею в потоці даних інформації про індекс предиктора, який на практиці використовується для прогнозування різниці векторів руху. Дивіться, наприклад, незатінений блок 122 на Фіг. 20. Реконструктор 404 може надавати різні предиктори для вектора руху цього блока, як, наприклад, прогнозуючи просторово вектор руху, як, наприклад, зліва, зверху, у формі комбінації попередніх двох напрямів і, таким чином, вперед, і тимчасово прогнозуючи вектор руху з вектора руху близько розташованої частини попередньо декодованої картинки відеозображення, і додаткові комбінації вищезгаданих предикторів. Ці предиктори зберігаються прогнозованим чином реконструктором 404, який передбачається на кодувальній стороні. Деяка інформація передається до цього кінця в потоці даних і використовується реконструктором. Тобто, деякий натяк міститься в потоці даних щодо того, який предиктор з цього упорядкованого списку предикторів слід реально використовувати як предиктор для вектора руху цього блока. Цей індекс може явно передаватися в потоці даних для цього блока. Однак, також можна, щоб індекс спершу прогнозувався, а потім просто передавався його прогноз. Також існують інші можливості. В будь-якому випадку, тільки що згадана схема прогнозування надає можливість дуже точного прогнозування вектора руху поточного блока і, відповідно, послаблюється вимога до інформаційного контенту, яка накладається на різницю векторів руху. Відповідно, обмеження адаптивного до контексту ентропійного кодування просто двома інформаційними величинами усиченого унарного коду і зниження критичної величини до 2, як описано по відношенню до Фіг. 18, а також вибір порядку експоненціального коду Голомба рівним 1, негативно не впливає на ефективність кодування, оскільки різниці векторів руху мають, завдяки високій ефективності прогнозування, частотну гістограму, згідно з якою більші величини компонентів 802x,у різниці векторів руху рідше використовуються. Навіть відмова від будь-якого розрізнення горизонтальних і вертикальних компонентів узгоджується з ефективним прогнозом, оскільки прогноз має тенденцію рівноцінно гарно використовуватися в обох напрямках з високою точністю. Суттєво відзначити, що у вищенаведеному описі усі деталі, вказані на Фіг. 1-15, також здатні переноситися на об'єкти, зображені на Фіг. 16, такі як, наприклад, настільки, наскільки це стосується функцій десимволізатора 314, реконструктора 404 і ентропійного декодера 409. Тим не менше, для повноти, деякі з цих деталей знову вказуються нижче.

Для кращого розуміння тільки що описаної схеми прогнозування, дивіться Фіг. 20. Як тільки що описано, конструктор 404 може одержувати різні предиктори для поточного блока 822 або поточної групи блоків для злиття, при цьому ці предиктори зображені векторами 824, зображеними суцільними лініями. Предиктори можуть одержуватися просторовим і/або тимчасовим прогнозуванням, де, додатково, операції арифметичного осереднення або подібне можуть використовуватися так, що окремі предиктори можуть одержуватися реконструктором 404 у такий спосіб, що вони корелюються між собою. Незалежно від способу одержання векторів 826 реконструктор 404 послідовно передає або сортує ці предиктори 126 в упорядкований список. Це вказано цифрами 1-4 на Фіг. 21. Бажано, щоб процес сортування був здатен унікально визначатися так, щоб кодер і декодер могли працювати синхронно. Потім, тільки що згаданий індекс може явно або неявно одержуватися реконструктором 404 для поточного блока або групи для злиття з потоку даних. Наприклад, другий предиктор "2" може вибиратися, а реконструктор 404 додає різницю векторів руху 800 до цього вибраного предиктора 126, таким чином надаючи врешті решт відновлений вектор руху 128, який потім використовується для прогнозування шляхом прогнозування з компенсацією руху контенту поточного блоку/групи для злиття. У випадку групи для злиття, повинно бути можливим, щоб

реконструктор 404 містив додаткові різниці векторів руху, передбачені для блоків групи для злиття, для подальшої деталізації вектора руху 128 по відношенню до окремих блоків групи для злиття.

Таким чином, продовжуючи далі описувати втілення об'єктів, зображених на Фіг. 16, можна, щоб ентропійний декодер 409 конфігурувався для одержання зрізаного унарного коду 806 з потоку даних 401 з використанням бінарного арифметичного декодування або бінарного PIPE кодування. Обидві концепції були описані вище. Окрім того, ентропійний декодер 409 може конфігуруватися для використання різних контекстів для двох положень інформаційної величини усіченого унарного коду 806 або, альтернативно, навіть однакового контексту для обох інформаційних величин. Ентропійний декодер 409 міг би конфігуруватися для виконання оновлення стану ймовірності. Ентропійний декодер 409 міг би робити це для інформаційної величини, поточно одержаної зі зрізаного унарного коду 806 шляхом переходу з поточного стану ймовірності, зв'язаного з контекстом, вибраним для поточно одержаної інформаційної величини, у новий стан ймовірності в залежності від поточно одержаної інформаційної величини. Дивіться вищенаведені таблиці `Next_State_LPS` і `Next_State_MPS`, таблиця пошуку для яких формується ентропійним декодером на додаток до інших вищеописаних етапів 0 - 5. У вищенаведеному обговоренні поточний стан ймовірності був позначений як `pState_current`. Він визначається для відповідного контексту, який представляє інтерес. Ентропійний декодер 409 може конфігуруватися для бінарного арифметичного декодування інформаційної величини, яка на даний момент одержується зі зрізаного унарного коду 806 шляхом дискретизації величини ширини поточного інтервалу ймовірності, тобто, `R`, яка представляє поточний інтервал ймовірності, для одержання індексу `q_index` інтервалу ймовірності, і шляхом виконання підрозбиття інтервалу шляхом індексації вхідної величини таблиці серед вхідних величин таблиці з використанням індексу інтервалу ймовірності і індексу стану ймовірності, тобто `p_state`, який, у свою чергу, залежить від поточного стану ймовірності, зв'язаного з контекстом, вибраним для інформаційної величини, яка поточно одержується, для одержання підрозбиття поточного інтервалу ймовірності на два часткові інтервали. У вищеописаних варіантах виконання ці часткові інтервали зв'язувалися з найбільш ймовірним і найменш ймовірним символом. Як описано вище, ентропійний декодер 409 може конфігуруватися для використання восьмибітового представлення для величини `R` ширини поточного інтервалу ймовірності з відбором, наприклад, двох або трьох найбільш значущих бітів восьмибітового представлення і дискретизацією величини ширини поточного інтервалу ймовірності. Ентропійний декодер 409 може додатково конфігуруватися для вибору серед двох часткових інтервалів на основі величини стану зміщення зсередини поточного інтервалу ймовірності, зокрема `V`, для оновлення величини `R` ширини поточного інтервалу ймовірності і величини стану зміщення, і для виведення значення інформаційної величини, яка поточно одержується, використовуючи вибраний частковий інтервал, і для виконання повторної нормалізації оновленої величини `R` ширини поточного інтервалу ймовірності і величини `V` стану зміщення, включаючи продовження зчитування бітів з потоку даних 401. Ентропійний декодер 409 може, наприклад, конфігуруватися для бінарного арифметичного декодування інформаційної величини з експоненціального коду Голомба шляхом поділу навпіл величини ширини поточного інтервалу ймовірності для одержання підрозбиття поточного інтервалу ймовірності на два часткові інтервали. Поділ навпіл відповідає оцінці ймовірності, яка фіксується рівною 0,5. Він може втілюватися простим зміщенням бітів. Ентропійний декодер може додатково конфігуруватися для кожної різниці векторів руху для одержання усіченого унарного коду горизонтальних і вертикальних компонентів відповідної різниці векторів руху з потоку даних 401 перед експоненціальним кодом Голомба горизонтальних і вертикальних компонентів відповідної різниці векторів руху. Завдяки цьому заходу ентропійний декодер 409 може використовувати таку більшу кількість інформаційних величин, які разом формують рядок інформаційних величин, для яких оцінка ймовірності фіксується, зокрема становить 0,5. Це може прискорювати процедуру ентропійного декодування. З іншого боку, ентропійний декодер 409 може надавати перевагу збереженню порядку серед різниць векторів руху шляхом, перш за все, одержання горизонтальних і вертикальних компонентів однієї різниці векторів руху, а потім просто продовженням одержувати горизонтальні і вертикальні компоненти наступної різниці векторів руху. Завдяки цьому заходу вимоги до пам'яті, які накладаються на декодувальний засіб, тобто, декодер з Фіг. 16, послаблюються, оскільки десимволізатор 314 може починати дебінаризацію різниць векторів руху негайно без очікування на перегляд подальших різниць векторів руху. Це дозволяється вибором контексту: оскільки просто саме один контекст доступний на положення інформаційної величини коду 806, просторовий взаємозв'язок не слід перевіряти.

Реконструктор 404 може, як описано вище, просторово і/або тимчасово прогнозувати горизонтальні і вертикальні компоненти векторів руху для одержання предикторів 126 для горизонтальних і вертикальних компонентів вектора руху і для відновлення горизонтальних і вертикальних компонентів векторів руху шляхом деталізації предикторів 826 з використанням

5 горизонтальних і вертикальних компонентів різниць векторів руху, як, наприклад, просто шляхом додавання різниці векторів руху до відповідного предиктора.

Окрім того, реконструктор 404 може конфігуруватися для прогнозування горизонтальних і вертикальних компонентів векторів руху різними способами для одержання упорядкованого списку предикторів для горизонтальних і вертикальних компонентів векторів руху, для

10 одержання індексу списку з потоку даних і для відновлення горизонтальних і вертикальних компонентів векторів руху шляхом деталізації предиктора, на який вказує індекс списку предикторів, з використанням горизонтальних і вертикальних компонентів різниць векторів руху.

Окрім того, як вже було описано вище, реконструктор 404 може конфігуруватися для відновлення відеоданих з використанням прогнозування з компенсацією руху шляхом застосування горизонтальних і вертикальних компонентів 802x,y векторів руху при просторовій

15 гранулярності, визначеній підрозбиттям відеокартинки в блоках, при цьому реконструктор 404 може використовувати синтаксичні елементи, які зливаються, присутні в потоці даних 401, для групування блоків в групи для злиття і для застосування цілих величин горизонтальних і вертикальних компонентів 802x,y різниць векторів руху, одержаних бінаризатором 314, в

20 елементах груп для злиття.

Реконструктор 404 може одержувати підрозбиття відеокартинки в блоках з частини потоку даних 401, який не включає синтаксичні елементи, які зливаються. Реконструктор 404 може також адаптувати горизонтальні і вертикальні компоненти наперед визначеного вектора руху для усіх блоків відповідної групи для злиття або деталізувати їх горизонтальними і

25 вертикальними компонентами різниць векторів руху, зв'язаних з блоками групи для злиття.

Тільки для повноти, Фіг. 17 зображає кодер, який підключається до декодера з Фіг. 16. Кодер з Фіг. 17 містить конструктор 504, символізатор 507 і ентропійний кодер 513.

Кодер містить конструктор 504, сконфігурований для прогнозованого кодування відеоданих 505 шляхом прогнозування з компенсацією руху, яке використовує вектори руху, і шляхом прогнозувального кодування векторів руху шляхом прогнозування векторів руху, і шляхом визначення цілих величин 506 горизонтальних і вертикальних компонентів різниць векторів руху для представлення похибки прогнозування зпрогнозованих векторів руху; символізатор 507, сконфігурований для бінаризації цілих величин для одержання результатів бінаризації 508

30 горизонтальних і вертикальних компонентів різниць векторів руху, при цьому результати бінаризації зрівнюють усічений унарний код горизонтальних і вертикальних компонентів, відповідно, в першому інтервалі області горизонтальних і вертикальних компонентів нижче критичної величини, і комбінацію префікса у формі зрізаного унарного коду для критичної величини і суфікса у формі експоненціального коду Голомба горизонтальних і вертикальних

35 компонентів, відповідно, в другому інтервалі області горизонтальних і вертикальних компонентів включно і з перевищенням критичної величини, де критична величина дорівнює двійці, а експоненціальний код Голомба має порядок одиниця; і ентропійний кодер 513, сконфігурований для кодування для горизонтальних і вертикальних компонентів різниць векторів руху зрізаного унарного коду з одержанням потоку даних з використанням адаптивного до контексту бінарного ентропійного кодування виключно з одним контекстом на положення інформаційної величини

40 усіченого унарного коду, який є спільним для горизонтальних і вертикальних компонентів різниць векторів руху, і експоненціального коду Голомба з використанням сталого рівномірного байпасного режиму. Подальші деталі можливого втілення безпосередньо переносяться з опису, який стосується декодера з Фіг. 16, до кодера з Фіг. 17.

Хоча деякі аспекти були описані в контексті пристрою, зрозуміло, що ці аспекти також

50 представляють опис відповідного способу, де блок або пристрій відповідає етапу способу або ознаці етапу способу. Аналогічно, аспекти, описані в контексті етапу способу, також представляють опис відповідного блоку або вузла, або ознаки відповідного пристрою. Деякі або усі етапи способу можуть виконуватися за допомогою (або використанням) апаратних засобів, як, наприклад, мікропроцесора, програмованого комп'ютера або електронної схеми. В деяких

55 варіантах виконання деякі або більша кількість найбільш важливих етапів способу можуть виконуватися таким пристроєм.

Кодований сигнал винаходу може зберігатися в середовищі для збереження цифрової інформації або може передаватися в передавальне середовище, таке як середовище безпроводної передачі даних або проводове передавальне середовище, таке як Інтернет.

В залежності від певних вимог для втілення, варіанти виконання винаходу можуть втілюватися в апаратних або в програмних засобах. Втілення може здійснюватися з використанням середовища для зберігання цифрової інформації, наприклад дискети, DVD, Blue-Ray, CD, ROM, PROM, EPROM, EEPROM або ФЛЕШ-пам'яті, які зберігають в собі електронно зчитувані керувальні сигнали, які взаємодіють (або здатні взаємодіяти) з програмованою комп'ютерною системою так, що виконується відповідний спосіб. Тому, дані з середовища для зберігання цифрової інформації можуть зчитуватися комп'ютером.

Деякі варіанти виконання згідно з винаходом містять носій даних, який містить електронно зчитувані керувальні сигнали, які здатні взаємодіяти з програмованою комп'ютерною системою так, що виконується один з описаних тут способів.

Загалом, варіанти виконання представленого винаходу можуть втілюватися як комп'ютерна програма з програмним кодом, який активується для виконання одного із способів, коли вона виконується на комп'ютері. Програмний код може, наприклад, зберігатися на зчитуваному машиною носії.

Інші варіанти виконання включають комп'ютерну програму для виконання одного з описаних тут способів, яка зберігається на зчитуваному машиною носії.

Іншими словами, варіант виконання способу винаходу є, тому, комп'ютерною програмою, яка має програмний код для виконання одного з описаних тут способів, коли вона виконується на комп'ютері.

Подальшим варіантом виконання винаходу є, тому, носій даних (або середовище для зберігання цифрової інформації або зчитуване комп'ютером середовище), який містить записану на собі комп'ютерну програму для виконання одного з описаних тут способів. Носій даних, середовище для зберігання цифрової інформації або середовище із записаною інформацією типово є матеріальними і/або неперехідними.

Подальший варіант виконання винаходу є, тому, потоком даних або послідовністю сигналів, які представляють комп'ютерну програму для виконання одного з описаних тут способів. Потік даних або послідовність сигналів може, наприклад, конфігуруватися для передачі за допомогою з'єднання зв'язку, наприклад по Інтернету.

Подальший варіант виконання включає засіб обробки, наприклад комп'ютер, або програмований логічний пристрій, сконфігурований або адаптований для виконання одного з описаних тут способів.

Подальший варіант виконання включає комп'ютер, який має встановлену на ньому комп'ютерну програму для виконання одного з описаних тут способів.

Подальший варіант виконання, згідно з винаходом, включає пристрій або систему, сконфігуровану для передачі (наприклад, електронно або оптично) комп'ютерної програми для виконання одного з описаних тут способів до приймача. Приймач може, наприклад, бути комп'ютером, мобільним пристроєм, запам'ятовуючим пристроєм або подібним. Пристрій або система може, наприклад, включати файловий сервер для передачі комп'ютерної програми до приймача.

В деяких варіантах виконання програмований логічний пристрій (наприклад, здатні до перепрограмування користувачем компоненти масиву) може використовуватися для виконання деяких або усіх функцій описаних тут способів. В деяких варіантах виконання здатні до перепрограмування користувачем компоненти масиву можуть взаємодіяти з мікропроцесором для виконання одного з описаних тут способів. Загалом, способи переважно виконуються будь-яким апаратним засобом.

Вищеописані варіанти виконання є просто ілюстрацією принципів представленого винаходу. Зрозуміло, що модифікації і варіанти описаних тут схем і деталей стануть очевидними для фахівців у цій галузі. Тому, наміром є обмеження об'єму правового захисту тільки наступною формулою винаходу, а не спеціальними деталями, представленими у вигляді опису і пояснення варіантів виконання.

ФОРМУЛА ВИНАХОДУ

1. Декодер для декодування відеоданих з потоку даних, у якому закодовані горизонтальні і вертикальні компоненти різниць векторів руху, з використанням, відповідно, бінаризації абсолютної величини горизонтальних і вертикальних компонентів, при цьому бінаризація містить префікс, який містить першу інформаційну величину з індексом 0, який вказує на те, чи перевищує абсолютна величина 0, чи ні, другу інформаційну величину з індексом 1, який вказує, чи перевищує абсолютна величина 1, чи ні, і яка просто кодується, коли перший ідентифікатор дорівнює 1, і, якщо абсолютна величина дорівнює або перевищує двійку, то суфікс у формі

- експоненціального коду Голомба порядку одиниця, який вказує залишок абсолютної величини в інтервалі включно і з перевищенням двійки, при цьому декодер містить ентропійний декодер, сконфігурований для одержання бінаризації різниць векторів руху для горизонтальних і вертикальних компонентів різниць векторів руху шляхом одержання першої і другої інформаційної величини з потоку даних з використанням адаптивного до контексту бінарного ентропійного декодування з точно одним контекстом для першої і другої інформаційної величини префіксу, який є, відповідно, спільним для горизонтальних і вертикальних компонентів різниць векторів руху, і експоненціального коду Голомба з використанням сталого рівномірного байпасного режиму;
- 5 десимволізатор, сконфігурований для дебінаризації результату бінаризації абсолютної величини для одержання абсолютної величини горизонтальних і вертикальних компонентів різниць векторів руху;
- 10 реконструктор, сконфігурований для відновлення відеоданих на основі абсолютної величини горизонтальних і вертикальних компонентів різниць векторів руху.
- 15 2. Декодер за п. 1, який **відрізняється** тим, що ентропійний декодер (409) сконфігурований для використання точно одного першого контексту для першої інформаційної величини префіксу, який є спільним для горизонтальних і вертикальних компонентів різниць векторів руху, і точно одного другого контексту для другої інформаційної величини префіксу, який є спільним для горизонтальних і вертикальних компонентів різниць векторів руху, при цьому перший і другий
- 20 контекст є різними.
3. Декодер за п. 1 або п. 2, який **відрізняється** тим, що ентропійний декодер (409) сконфігурований для виконання оновлення стану ймовірності для інформаційної величини, поточно одержаної з префіксу (806), шляхом переходу з поточного стану ймовірності, зв'язаного з контекстом, вибраним для поточно одержаної інформаційної величини, до нового стану ймовірності в залежності від поточно одержаної інформаційної величини.
- 25 4. Декодер за будь-яким із пп. 1-3, який **відрізняється** тим, що потік даних має кодовану в нього карту глибини.
5. Декодер за будь-яким із пп. 1-3, який **відрізняється** тим, що потік даних має кодовані в нього відеодані у вигляді масиву інформаційних зразків, які відповідають значенням кольору.
- 30 6. Кодер для кодування відеоданих з одержанням потоку даних, який містить конструктор, сконфігурований для прогнозованого кодування відеоданих за допомогою прогнозування з компенсацією руху з використанням векторів руху і шляхом прогнозованого кодування векторів руху шляхом прогнозування векторів руху, і шляхом визначення цілих величин горизонтальних і вертикальних компонентів різниць векторів руху для представлення похибки прогнозування
- 35 спрогнозованих векторів руху;
- символізатор, сконфігурований для бінаризації абсолютної величини горизонтальних і вертикальних компонентів різниць векторів руху для одержання результату бінаризації абсолютної величини горизонтальних і вертикальних компонентів різниць векторів руху, при цьому бінаризація містить префікс, який містить першу інформаційну величину з індексом 0,
- 40 який вказує на те, чи перевищує абсолютна величина 0, чи ні, другу інформаційну величину з індексом 1, який вказує, чи перевищує абсолютна величина 1, чи ні, і яка просто кодується, коли перший ідентифікатор дорівнює 1, і, якщо абсолютна величина дорівнює або перевищує двійку, то суфікс у формі експоненціального коду Голомба порядку одиниця, який вказує залишок абсолютної величини в інтервалі включно і з перевищенням двійки; і
- 45 ентропійний кодер, сконфігурований для кодування для горизонтальних і вертикальних компонентів різниць векторів руху першої і другої інформаційної величини з одержанням потоку даних з використанням адаптивного до контексту бінарного ентропійного кодування з точно одним контекстом для першої і другої інформаційної величини префіксу, який є, відповідно, спільним для горизонтальних і вертикальних компонентів різниць векторів руху, і
- 50 експоненціального коду Голомба з використанням сталого рівномірного байпасного режиму.
7. Кодер за п. 6, який **відрізняється** тим, що ентропійний кодер сконфігурований для використання точно одного першого контексту для першої інформаційної величини префіксу, який є спільним для горизонтальних і вертикальних компонентів різниць векторів руху, і точно одного другого контексту для другої інформаційної величини префіксу, який є спільним для
- 55 горизонтальних і вертикальних компонентів різниць векторів руху, при цьому перший і другий контекст є різними.
8. Кодер за п. 6 або п. 7, який **відрізняється** тим, що ентропійний кодер сконфігурований для виконання оновлення стану ймовірності для інформаційної величини, поточно одержаної з префіксу, шляхом переходу з поточного стану ймовірності, зв'язаного з контекстом, вибраним

для поточно одержаної інформаційної величини, до нового стану ймовірності в залежності від поточно одержаної інформаційної величини.

9. Кодер за будь-яким із пп. 6-8, який **відрізняється** тим, що сконфігурований для кодування в потік даних карти глибини.

5 10. Кодер за будь-яким із пп. 6-8, який **відрізняється** тим, що сконфігурований для кодування в потік даних відеоданих у вигляді масиву інформаційних зразків, які відповідають значенням кольору

11. Машинозчитуваний носій інформації, який зберігає потік даних, який має відеодані, кодовані в нього з використанням прогнозування з компенсацією руху з використанням векторів руху і шляхом прогнозованого кодування векторів руху шляхом прогнозування векторів руху, при цьому потік даних сигналізує цілі величини горизонтальних і вертикальних компонентів різниць векторів руху для представлення похибки прогнозування спрогнозованих векторів руху, при цьому абсолютна величина горизонтальних і вертикальних компонентів різниць векторів руху кодується в потік даних у формі бінаризації абсолютної величини горизонтальних і вертикальних компонентів різниць векторів руху, при цьому бінаризація містить префікс, який містить першу інформаційну величину з індексом 0, який вказує на те, чи перевищує абсолютна величина 0, чи ні, другу інформаційну величину з індексом 1, який вказує, чи перевищує абсолютна величина 1, чи ні, і яка просто кодується, коли перший ідентифікатор дорівнює 1, і, якщо абсолютна величина дорівнює або перевищує двійку, то суфікс у формі експоненціального коду Голомба порядку одиниця, який вказує залишок абсолютної величини в інтервалі включно і з перевищенням двійки; і при цьому перша і друга інформаційна величина ентропійно кодуються з одержанням потоку даних з використанням адаптивного до контексту бінарного ентропійного кодування з точно одним контекстом для першої і другої інформаційної величини префіксу, який є, відповідно, спільним для горизонтальних і вертикальних компонентів різниць векторів руху, і експоненціальний код Голомба ентропійно кодується в потік даних з використанням сталого рівномірного байпасного режиму.

12. Машинозчитуваний носій інформації за п. 11, який **відрізняється** тим, що має кодовану в нього карту глибини.

13. Машинозчитуваний носій інформації за п. 11, який **відрізняється** тим, що має кодовані в нього відеодані у вигляді масиву інформаційних зразків, які відповідають значенням кольору.

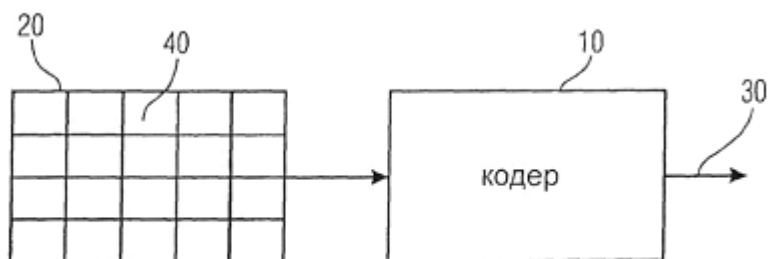


Fig. 1

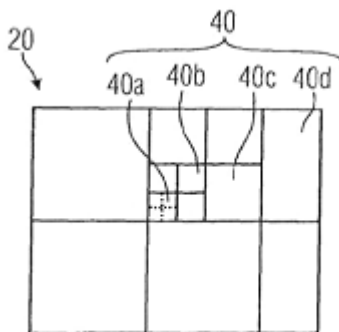


Fig. 2 A

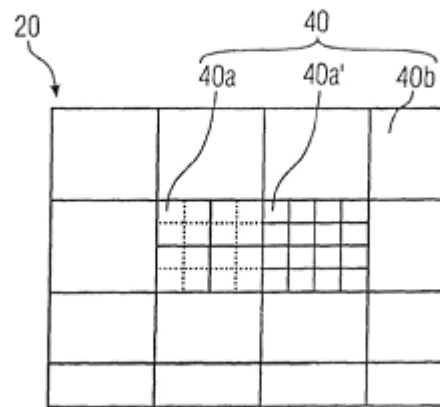


Fig. 2 B

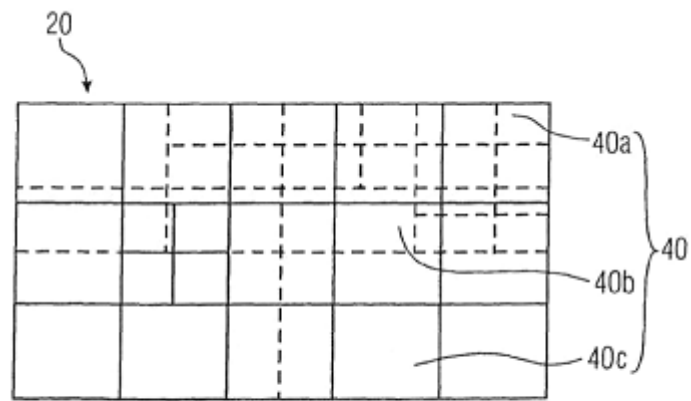


Fig. 2 C

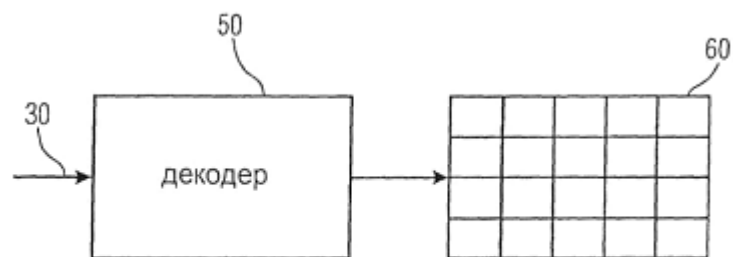
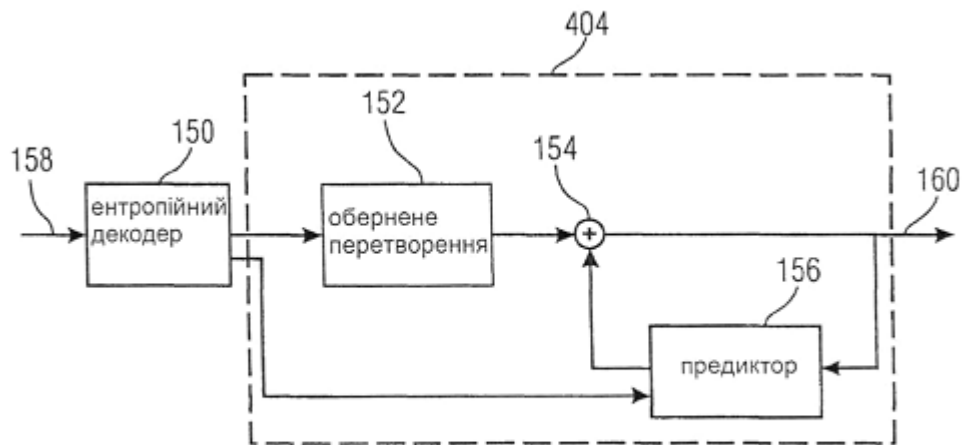


Fig. 3



Фиг. 4



Фиг. 5

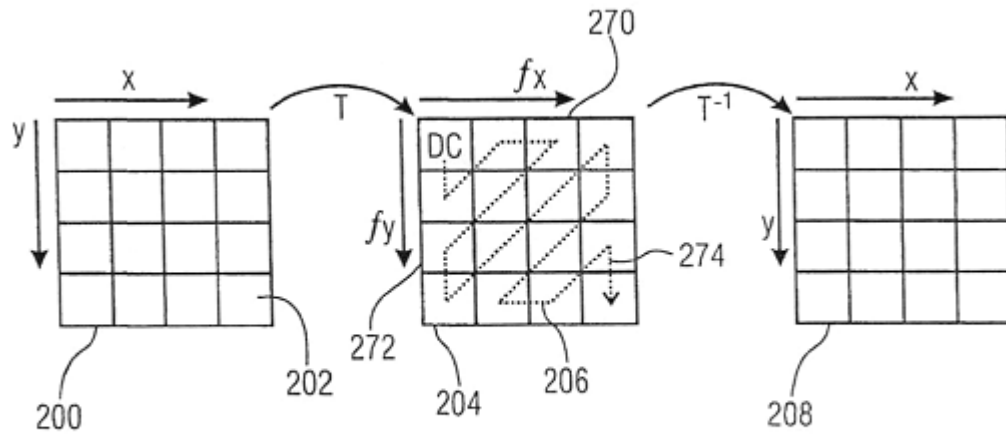


Fig. 6

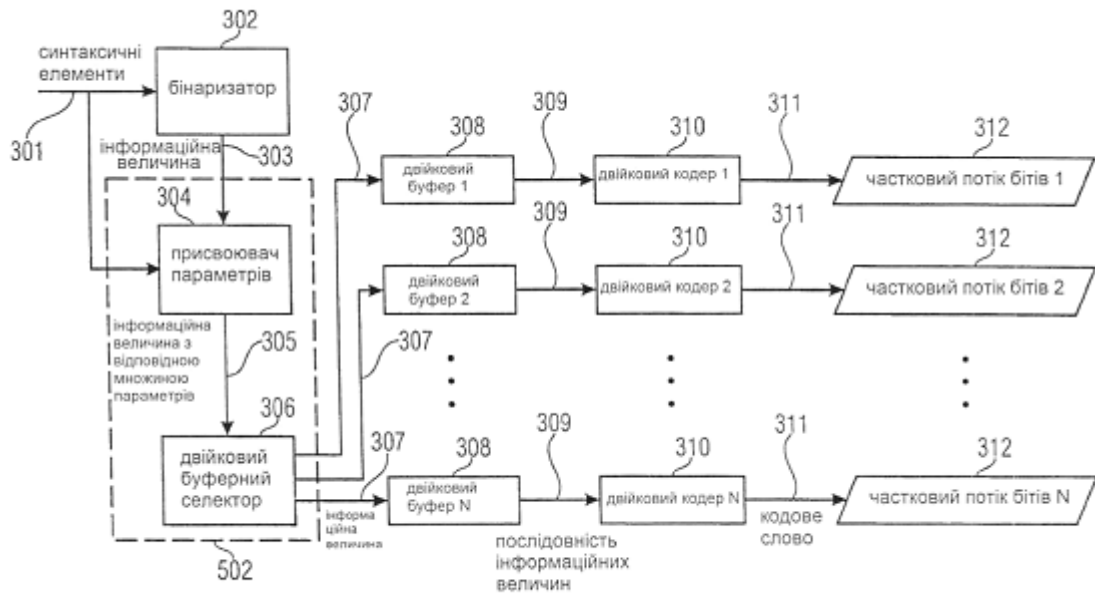


Fig. 7

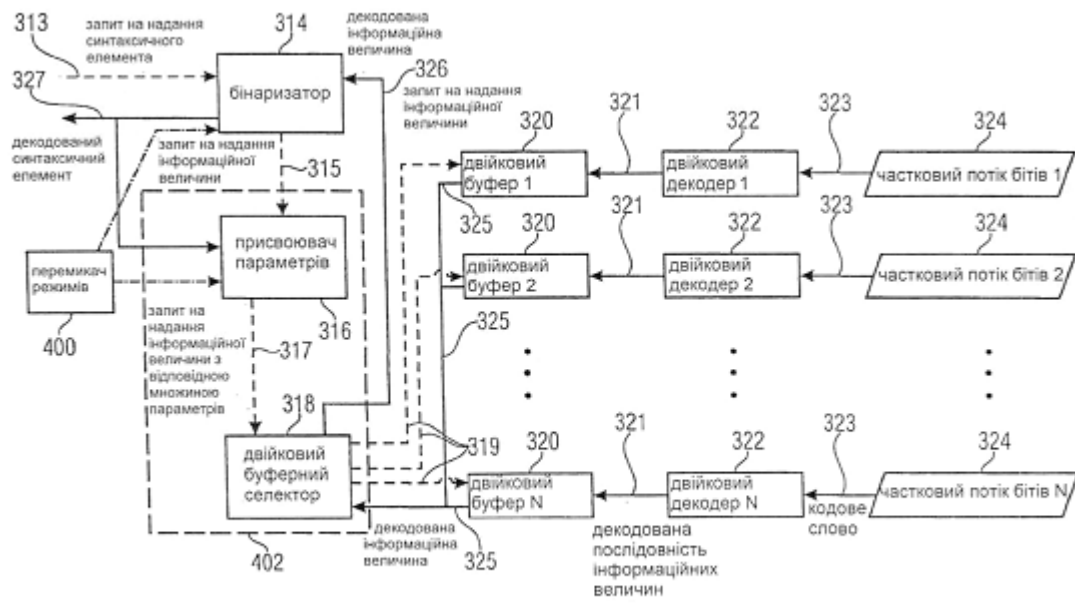


Fig. 8

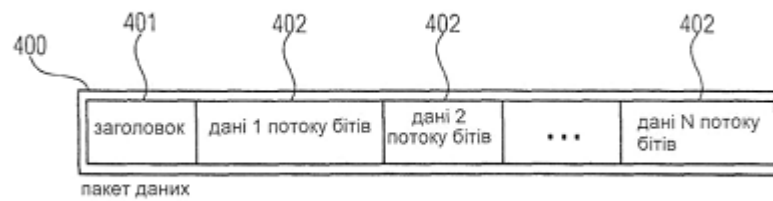


Fig. 9

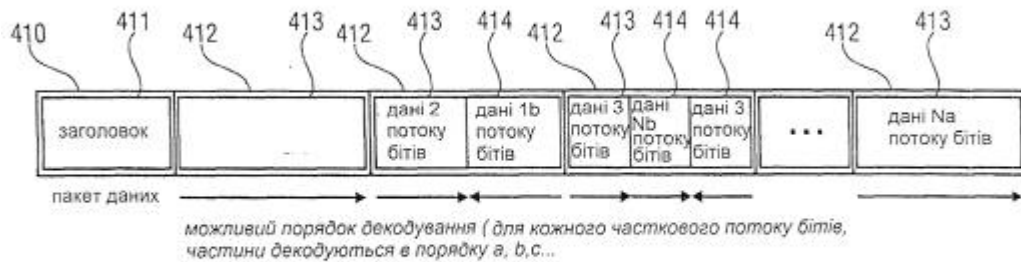
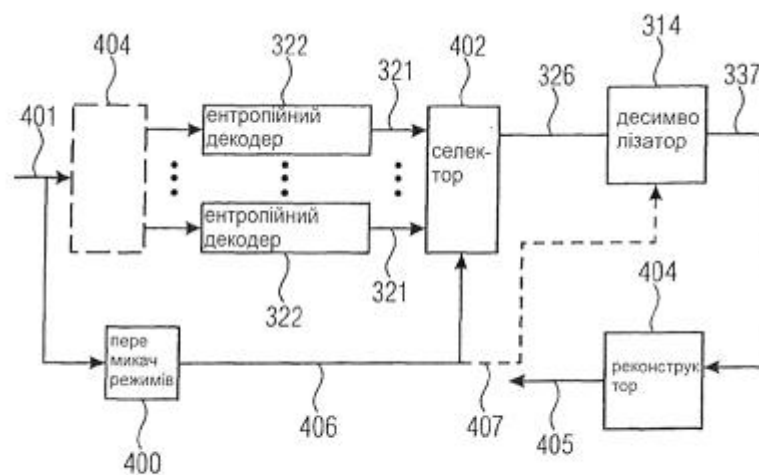
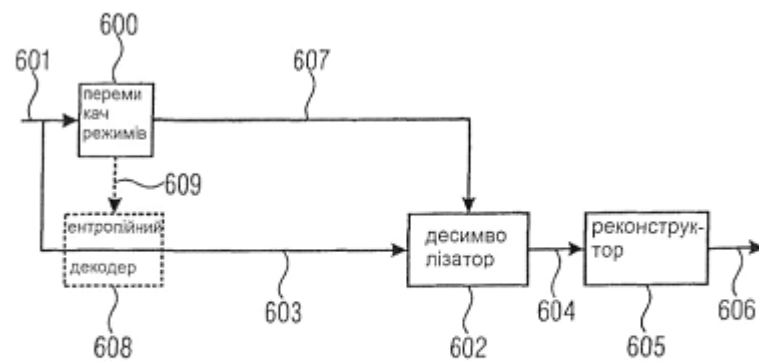


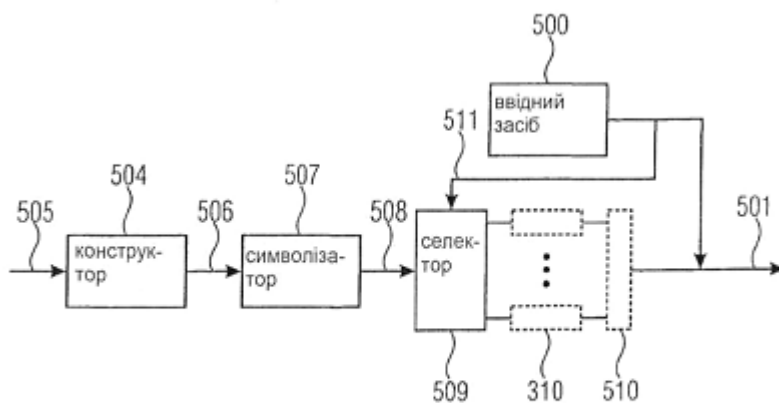
Fig. 10



Фиг. 11



Фиг. 12



Фиг. 13

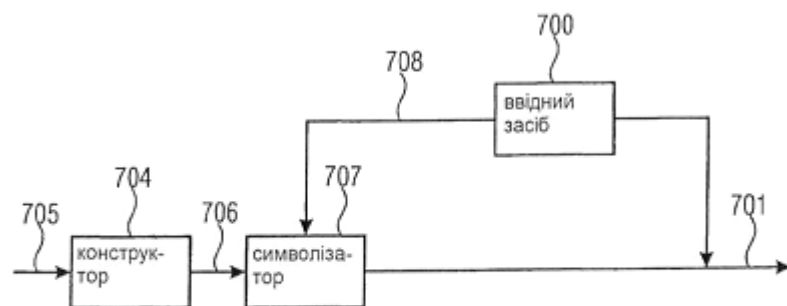


Fig. 14

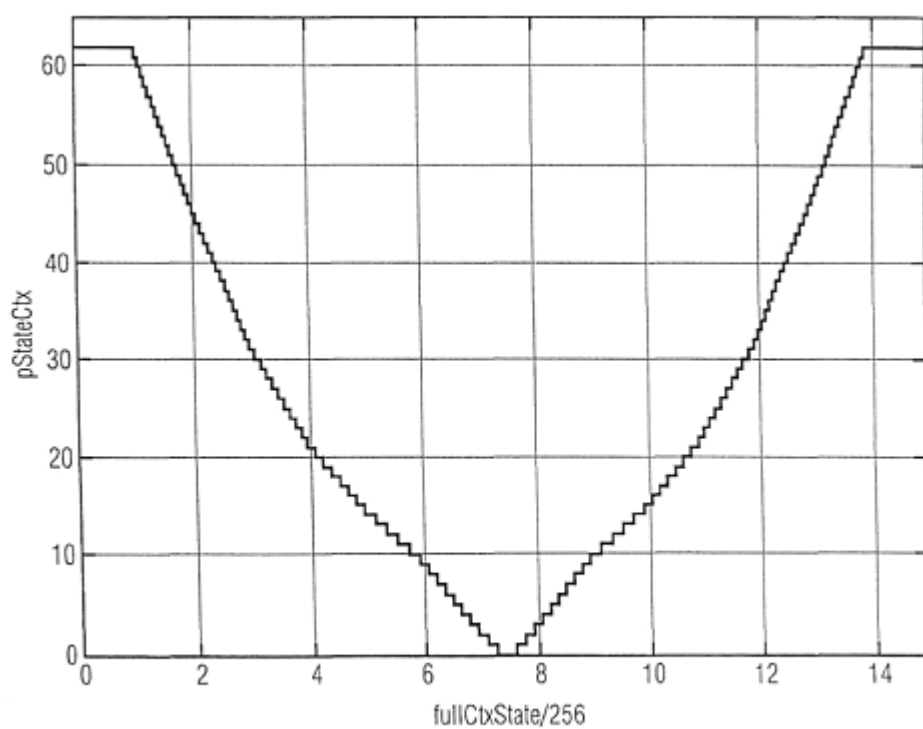


Fig. 15



Fig. 16

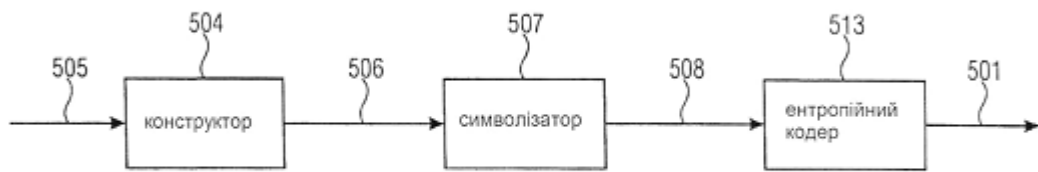


Fig. 17

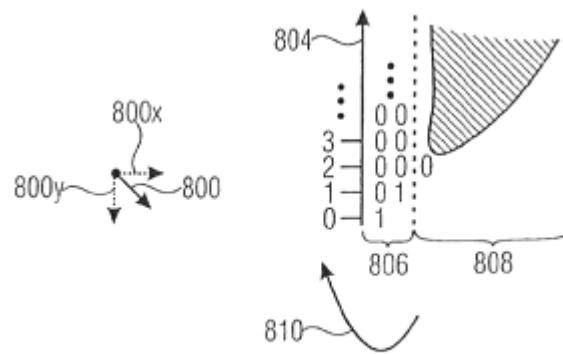


Fig. 18

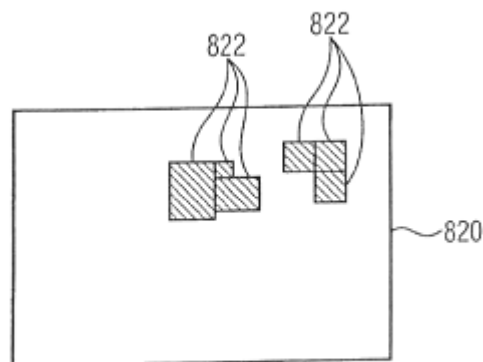


Fig. 19

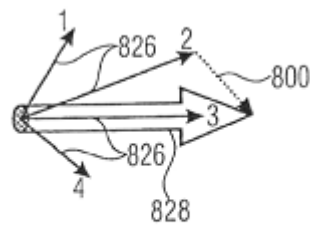


Fig. 20

Комп'ютерна верстка В. Мацело

Міністерство економічного розвитку і торгівлі України, вул. М. Грушевського, 12/2, м. Київ, 01008, Україна

ДП "Український інститут інтелектуальної власності", вул. Глазунова, 1, м. Київ – 42, 01601