РОССИЙСКАЯ ФЕДЕРАЦИЯ

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

(19) **RU** (11) **2 595 916** (13) **C1**

(51) МПК
H03M 7/32 (2006.01)
H03M 7/36 (2006.01)

(12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

(21)(22) Заявка: **2015132041/08, 27.02.2014**

(24) Дата начала отсчета срока действия патента:
27.02.2014

Приоритет(ы):
(30) Конвенционный приоритет:
01.03.2013 GB 1303661.1

(45) Опубликовано: **27.08.2016** Бюл. № **24**

(56) Список документов, цитированных в отчете о поиске: **EP 2214315 A1, 04.08.2010. EP 1376974 A1, 02.01.2004. US 6642874 B1, 04.11.2003. US 2012141040 A1, 07.06.2012. SU 1762411 A1, 15.09.1992. SU 1499501 A1, 07.08.1989.**

(85) Дата начала рассмотрения заявки PCT на национальной фазе: **01.10.2015**

(86) Заявка PCT:
EP 2014/000510 (27.02.2014)

(87) Публикация заявки PCT:
WO 2014/131517 (04.09.2014)

Адрес для переписки:
191036, Санкт-Петербург, а/я 24, "НЕВИНПАТ"

(72) Автор(ы):
КАЛЕВО Осси (FI)

(73) Патентообладатель(и):
Гуруложик Микросистемс Ой (FI)

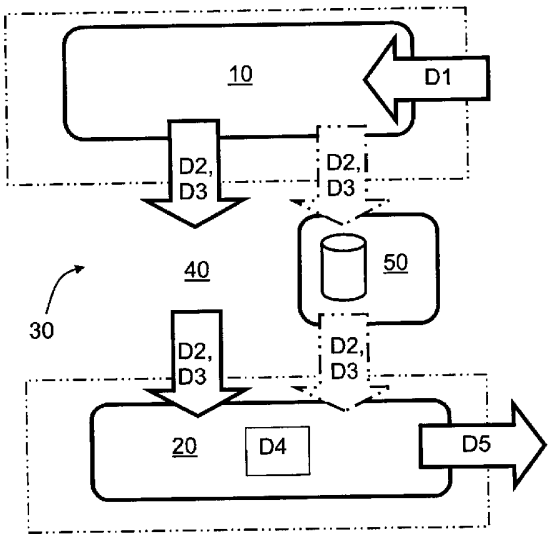
(54) КОДЕР, ДЕКОДЕР И СПОСОБ

(57) Реферат:

Группа изобретений относится к области кодирования. Техническим результатом является увеличение степени сжатия данных. Кодек (30) включает, по меньшей мере, один кодер (10) и, по меньшей мере, один декодер (20). Кодер включает схему обработки данных для применения к входным данным (D1) одной из форм разностного и/или суммирующего кодирования для формирования одной или более соответствующих кодированных последовательностей, которую подвергают циклическому переходу относительно максимального значения и/или циклическому переходу относительно минимального значения для формирования кодированных выходных данных (D2 или D3). Декодер включает схему обработки данных для обработки одной или более частей кодированных данных (D2 или D3), выполненную с возможностью применения одного из видов разностного и/или суммирующего декодирования к одной или более соответствующих кодированных последовательностей упомянутых одной или более частей, при этом одна или более кодированные последовательности подвергаются операции циклического перехода относительно максимального значения и/или циклического перехода относительно минимального значения, для формирования декодированных выходных данных (D5). 7 н. и 37 з.п. ф-лы, 3 ил., 2 табл.

RU 2595916 C1

RU 2595916 C1



Фиг.1

RU 2595916 C1

RU 2595916 C1

RUSSIAN FEDERATION

FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY

(19) **RU** (11) 2 595 916 (13) **C1**

(51) Int. Cl.
H03M 7/32 (2006.01)
H03M 7/36 (2006.01)

(12) ABSTRACT OF INVENTION

(21)(22) Application: **2015132041/08, 27.02.2014**

(24) Effective date for property rights:
27.02.2014

Priority:
(30) Convention priority:
01.03.2013 GB 1303661.1

(45) Date of publication: **27.08.2016** Bull. № **24**

(85) Commencement of national phase: **01.10.2015**

(86) PCT application:
EP 2014/000510 (27.02.2014)

(87) PCT publication:
WO 2014/131517 (04.09.2014)

Mail address:
191036, Sankt-Peterburg, a/ja 24, "NEVINPAT"

(72) Inventor(s):
KALEVO Ossi (FI)

(73) Proprietor(s):
Gurulodzhik Mikrosistems Oj (FI)

RU 2595916 C1

(54) ENCODER, DECODER AND METHOD

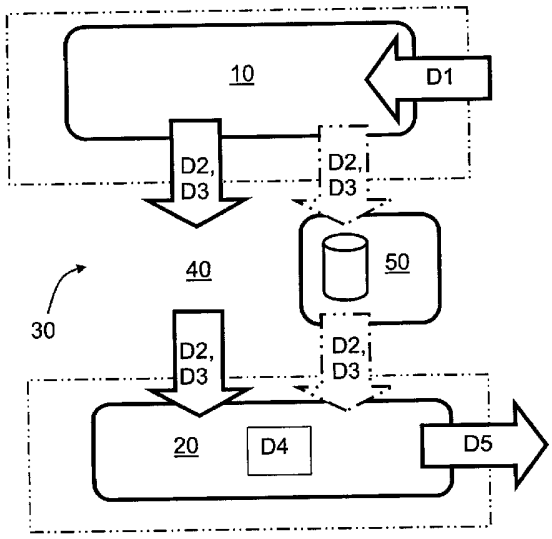
(57) Abstract:

FIELD: information technology.

SUBSTANCE: codec (30) includes at least one coder (10) and at least one decoder (20). Encoder includes data processing circuit for application to input data (D1) of one of forms of differential and/or summing coding to form one or more corresponding coded sequences, which is subjected to cyclic shift relative to maximum value and/or cyclic shift relative to minimum value to generate encoded output data (D2 or D3). Decoder includes data processing circuit for processing one or more parts of encoded data (D2 or D3) configured to use one of difference and/or summing decoding types to one or more corresponding coded sequences specified one or more parts, wherein one or more encoded sequences are subjected to cyclic transition operation relative to maximum value and/or cyclic transition relative to minimum value for formation of decoded output data (D5).

EFFECT: high degree of data compression.

44 cl, 3 dwg, 2 tbl



Фиг.1

RU 2595916 C1

Область техники

Настоящее изобретение относится к кодерам, например, к кодерам, которые подходят для применения оператора прямого О-дельта-кодирования (О-дельта-кодирования). Также настоящее изобретение относится к способам кодирования данных, например, к способам кодирования данных с применением оператора прямого О-дельта-кодирования. Также настоящее изобретение относится к декодерам для декодирования кодированных данных, например, к декодерам, которые подходят для применения оператора обратного О-дельта-кодирования. Дополнительно, настоящее изобретение относится к способам декодирования кодированных данных, например, к способам декодирования кодированных данных с применением оператора обратного О-дельта-кодирования. Также, дополнительно, настоящее изобретение относится к программным продуктам, записанным на машиночитаемом носителе для хранения данных, при этом такие программные продукты могут исполняться на вычислительном оборудовании с целью реализации упомянутых выше способов.

Предпосылки создания изобретения

Математическую теорию связи, которая легла в основу современных систем связи, предложил Клод Шеннон. При этом в свете упомянутой теории были разработаны различные современные методы кодирования. Список информационных источников, дающих обзор современного уровня технических знаний, приведен в таблице 1.

Таблица 1: Текущий уровень техники

Документ с описанием уровня техники	Подробная информация
P1	«Кодирование переменной длины» (“Variable-length code”), Википедия (по состоянию на 28 ноября 2012 года) URL: http://en.wikipedia.org/wiki/Variable-length_code

P2	«Кодирование длин серий», Википедия (по состоянию на 28 ноября 2012 года) URL: http://en.wikipedia.org/wiki/Run-length_encoding
P3	«Кодирование Хаффмана», Википедия (по состоянию на 28 ноября 2012 года) URL: http://en.wikipedia.org/wiki/Huffman_coding
P4	«Арифметическое кодирование», Википедия (по состоянию на 28 ноября 2012 года) URL: http://en.wikipedia.org/wiki/Arithmetic_coding
P5	«Математическая теория связи» ("A Mathematic Theory of Communication"), Клод Шеннон, 1948 год (по состоянию на 28 ноября 2012 года) URL: http://cm:bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf
P6	«Дельта-кодирование», Википедия (по состоянию на 28 ноября 2012 года) URL: http://en.wikipedia.org/wiki/Delta_coding
P7	Теорема Шеннона о кодировании источника; Википедия (по состоянию на 28 ноября 2012 года) URL: http://en.wikipedia.org/wiki/Source_coding_theorem
P8	«Энтропия» – Википедия (по состоянию на 28 ноября 2012 года) URL: http://en.wikipedia/wiki/Entropy

В патенте EP 1376974 B1 («Способ и устройство для сжатия заголовка пакета» ("Method and apparatus for packet header compression"), принадлежащем компании Alcatel Lucent, Франция) описан способ передачи пакетов данных, в котором пакеты данных включают поле (CF), содержащее сжатое значение. Это сжатое значение представляет собой значение, изменяющееся от пакета к пакету и охватывающее заранее заданный интервал (далее - «интервал интерпретации»). Способ включает следующие шаги:

(i) если расстояние между сжимаемым значением и заранее заданной границей циклического перехода меньше, чем заранее заданное пороговое значение, добавление дополнительного бита к сжатому значению, причем этот дополнительный бит уникально задает положение сжатого значения относительно границы циклического перехода;

(ii) в приемнике, вычисление первого интервала интерпретации в соответствии со всеми принятыми битами сжатого значения;

(iii) сигнализация о циклическом переходе, если более одного значения в первом интервале интерпретации совпадает с принятым сжатым значением;

(iv) после сигнализации о циклическом переходе, вычисление второго интервала интерпретации в соответствии со всеми принятыми битами сжатого значения, кроме дополнительного бита; и

(v) разрешение неопределенности декомпрессированного значения во втором интервале интерпретации с использованием дополнительного бита.

Определение энтропии Шеннона дано в документах P7 и P8, перечисленных в таблице 1. Существует множество различных методов сжатия, которые могут применяться для уменьшения энтропии исходных данных. Такие методы иногда применяют для изменения энтропии, например, для получения более высоких степеней сжатия исходных данных, выполняемого без потерь; такие изменяющие энтропию методы включают, например, кодирование длин серий (run-length-encoding, RLE) в соответствии с описанием в документе P2 из таблицы 1, кодирование переменной длины (variable-length-coding, VLC) в соответствии с описанием в документе P1 из таблицы 1, кодирование Хаффмана в соответствии с описанием в документе P3 из таблицы 1, дельта-кодирование в соответствии с описанием в документе P4 из таблицы 1. Эти методы могут эффективно применяться для сжатия данных, представляющих собой алфавитные символы, числа, байты или слова. Однако эти методы не являются хорошо приспособленными для сжатия исходных данных на битовом уровне, и по этой причине не подходят для эффективного сжатия исходных данных, меняющихся побитно.

Дельта-кодирование, например, в соответствии с описанием в документе P6 из таблицы 1, подходит для формирования дельта-значений (значений разности), которые могут быть положительными или отрицательными значениями, на основе положительных значений исходных данных. При этом известны существующие реализации дельта-кодеров для применения с 8-битным, 16-битным или 32-битным циклическим переходом, в зависимости от применяемого размера элементов данных. Однако не существует современных дельта-кодеров, оптимизированных для других режимов, помимо 8-битных, 16-битных или 32-битных значений. В частности, существующие дельта-кодеры особенно неэффективны при кодировании исходных битовых значений, а именно «0» и «1», например, при побитовом кодировании, так как в результате их работы обычно получают три различных значения, а именно «-1», «0» и «1».

Любые типы данных занимают определенный объем памяти, а при перемещении данных из одного места в другое необходимы передающие ресурсы системы связи. С ростом объемов данных, например, в результате развития мультимедийных технологий, таких как 3-мерный видеоконтент, соответственно, требуется все больший объем памяти и все больше передающих ресурсов для обращения с данными, и при этом также, с ростом объемов данных, растет энергопотребление. На глобальном уровне, объемы передаваемых данных увеличиваются с каждым днем; например, Интернет содержит громадные объемы данных, часть из которых хранится во множестве копий. При этом на существующем уровне техники известны методы понижения энтропии E, связанной с данными, например, для применения при сжатии объема данных. Также на существующем уровне техники известны способы изменения энтропии, к примеру, дельта-кодирование или кодирование длин серий (RLE), однако необходимы усовершенствованные методы для обеспечения большей степени сжатия данных, чем это возможно на сегодняшний день.

Также требуется оптимизация применения существующих методов дельта-кодирования, которая позволила бы более быстро и более эффективно выполнять кодирование исходных данных, например, побитное кодирование, при котором

используются не все значения из элементов исходных данных, и/или при котором предыдущий или последующий метод кодирования, применяемый в комбинации с дельта-кодированием, требует более формата большей разрядности, нежели побитное изменение, исходно применяемое для кодируемых данных.

Сущность изобретения

Цель настоящего изобретения - предложить усовершенствованную форму дельта-кодера, а именно «прямой O-дельта-кодер», который более эффективен при кодировании индивидуальных битов, а именно, при побитном кодировании, а также для других значений данных.

Еще одна цель настоящего изобретения - предложить усовершенствованный способ дельта-кодирования данных, а именно, способ прямого O-дельта-кодирования данных.

Еще одна цель настоящего изобретения - предложить усовершенствованный декодер для декодирования кодированных данных, например, данных, закодированных методом O-дельта-кодирования, а именно, обратный O-дельта-декодер.

Еще одна цель настоящего изобретения - предложить усовершенствованный способ декодирования кодированных данных, например, данных, закодированных методом O-дельта-кодирования, а именно, способ обратного O-дельта-декодирования данных.

В соответствии с первым аспектом настоящего изобретения предложен кодер по п. 1 приложенной формулы изобретения: предложен кодер для кодирования входных данных (D1), включающих последовательность числовых значений, с целью формирования соответствующих кодированных выходных данных (D2 или D3), включающий схему обработки данных для применения ко входным данным (D1) одной из форм разностного и/или суммирующего кодирования с целью формирования одной или более соответствующих кодированных последовательностей, при этом одну или более соответствующих кодированных последовательностей подвергают циклическому переходу (wraparound) относительно максимального значения и/или циклическому переходу относительно минимального значения с целью формирования кодированных выходных данных (D2 или D3).

Предпочтительно, циклический переход применяют для исключения появления отрицательных разностей или слишком больших значений при реализации кодера. Выражение «числовое значение» следует понимать как включающее индивидуальные биты (двоичное значение), а также группы бит (значение большего порядка, чем двоичное).

Настоящее изобретение обладает тем преимуществом, что комбинация разностного и/или суммирующего кодирования с циклическим переходом позволяет получить полезное изменение энтропии входных данных (D1) с формированием кодированных данных (D2), что обеспечивает возможность более эффективного сжатия данных в заданном энтропийном кодере при формировании кодированных данных (D3) на основе кодированных данных (D2).

Опционально, для кодера, схема обработки данных выполнена с возможностью анализа входных данных (D1) и/или одной или более соответствующих кодированных последовательностей с целью вычисления одного или более значений смещения, минимальных и/или максимальных значений для применения к одной или более соответствующим кодированным последовательностям с целью использования при формировании кодированных выходных данных (D2 или D3). Также, опционально, для кодера, одно или более значений смещения имеют значение «0».

Опционально, кодер выполнен с возможностью обработки числовых значений, включающих одно или более 1-битных значений, и при этом кодер выполнен с

возможностью кодирования входных данных (D1) побитно.

Опционально, в кодере, одна или более соответствующих кодированных последовательностей представляют изменения в последовательных значениях входных данных (D1).

Опционально, кодер выполнен с возможностью применения значения циклического перехода (wrapValue), при этом значение циклического перехода равно максимальному значению (highValue) минус минимальное значение (lowValue) плюс 1. Это значение циклического перехода используют, предпочтительно, при формировании кодированных выходных данных (D2 или D3), чтобы исключить появление таких значений, которые были бы меньше минимального значения (lowValue) (т.е. часто, отрицательным значением) или больше максимального значения (highValue).

Опционально, кодер выполнен с возможностью разбиения входных данных (D1) на множество фрагментов данных, которые кодируют отдельно. Также, опционально, кодер выполнен с возможностью избирательного применения кодирования к фрагментам данных, только когда при помощи этого может быть достигнуто сжатие данных в кодированных выходных данных (D2 или D3).

Опционально, кодер выполнен с возможностью применения заданного по умолчанию первого значения предсказания для последовательности значений предсказания, которые применяют для создания выходных кодированных данных (D2 или D3), с использованием входного значения, значения предсказания и оператора кодирования. Также, опционально, в кодере, заданное по умолчанию первое значение предсказания равно «0». Также, опционально, первое значение предсказания может представлять собой минимальное значение (lowValue), максимальное значение (highValue), (максимальное значение (highValue) плюс минимальное значение (lowValue) плюс 1) разделить на 2, и т.п.

Опционально, кодер выполнен с возможностью применения дополнительного кодирования с целью формирования кодированных выходных данных (D2 или D3), при этом дополнительное кодирование включает по меньшей мере одно из следующего: кодирование длин серий (RLE), раздельное RLE-кодирование, модификатор энтропии (entropy modifier, EM), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование.

Опционально, кодер выполнен с возможностью разбиения входных данных (D1) на множество фрагментов в зависимости от длины серий идентичных друг другу битов в них, что является эффективным для кодирования с использованием кодирования длин серий (RLE), раздельного RLE-кодирования, модификатора энтропии (EM), кодирования Хаффмана, кодирования переменной длины (VLE), интервального кодирования и/или арифметического кодирования.

Опционально, в кодере схема обработки данных реализована с использованием вычислительной аппаратуры, выполненной с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

В соответствии со вторым аспектом настоящего изобретения предложен способ использования кодера для кодирования входных данных (D1), включающих последовательность числовых значений, с целью формирования соответствующих кодированных выходных данных (D2 или D3), включающий:

(a) использование схемы обработки данных кодера для применения ко входным данным (D1) одной из форм разностного и/или суммирующего кодирования, с целью получения одной или более соответствующих кодированных последовательностей; и

(b) использование схемы обработки данных для выполнения над одной или более соответствующими кодированными последовательностями операции циклического перехода относительно максимального значения и/или циклического перехода относительно минимального значения, с целью формирования кодированных выходных данных (D2 или D3).

Предпочтительно, циклический переход применяют для исключения появления отрицательных разностей или слишком больших значений при реализации способа кодирования. Выражение «числовое значение» следует понимать как включающее индивидуальные биты (двоичное значение), а также группы битов (значение большего порядка, чем двоичное).

Опционально, способ включает использование схемы обработки данных для анализа входных данных (D1) и/или одной или более соответствующих кодированных последовательностей с целью вычисления одного или более значений смещения, минимальных и/или максимальных значений для применения к одной или более соответствующим кодированным последовательностям с целью использования при формировании кодированных выходных данных (D2 или D3). Также, опционально, способ включает использование, в качестве одного или более значений смещения, значения «0».

Опционально, способ включает обработку числовых значений, включающих одно или более 1-битных значений, при этом кодер выполнен с возможностью кодирования входных данных (D1) побитно. Также, опционально, способ включает использование, в качестве одного или более значений смещения, значения «0».

Опционально, при реализации способа, одна или более соответствующих кодированных последовательностей представляют изменения в последовательных значениях входных данных (D1).

Опционально, способ включает управление кодером для применения значения циклического перехода (wrapValue), при этом значение циклического перехода равно максимальному значению (highValue) минус минимальное значение (lowValue) плюс 1. Это значение циклического перехода используют, предпочтительно, при формировании кодированных выходных данных (D2 или D3), чтобы исключить появление таких значений, которые были бы меньше минимального значения (lowValue) (т.е. часто, отрицательным значением) или больше максимального значения (highValue).

Опционально, способ включает разбиение входных данных (D1) на множество фрагментов данных, которые кодируют отдельно. Также, опционально, способ включает избирательное применение кодирования к фрагментам данных, только когда при помощи этого может быть достигнуто сжатие данных в кодированных выходных данных (D2 или D3).

Опционально, способ включает использование кодера для применения заданного по умолчанию первого значения предсказания для последовательности значений предсказания, которые применяют для создания кодированных выходных данных (D2 или D3), с использованием входного значения, значения предсказания и оператора кодирования. Также, опционально, в этом способе заданное по умолчанию первое значение предсказания равно «0». Также, опционально, первое значение предсказания может представлять собой минимальное значение (lowValue), максимальное значение (highValue), (максимальное значение (highValue) плюс минимальное значение (lowValue) плюс 1) разделить на 2, и т.п.

Опционально, способ включает применение дополнительного кодирования с целью формирования кодированных выходных данных (D2 или D3), при этом дополнительное

кодирование включает по меньшей мере одно из следующего: кодирование длин серий (RLE), раздельное RLE-кодироавние, модификатор энтропии (EM), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование.

Также, опционально, способ включает разбиение входных данных (D1) на множество фрагментов в зависимости от длины серий идентичных друг другу битов в них, что является эффективным для кодирования с использованием кодирования длин серий (RLE), раздельного RLE-кодирования, модификатора энтропии (EM), кодирования Хаффмана, кодирования переменной длины (VLE), интервального кодирования и/или арифметического кодирования.

Также, опционально, способ включает реализацию схемы обработки с использованием вычислительной аппаратуры, выполненной с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

В соответствии с третьим аспектом настоящего изобретения предложен декодер для декодирования кодированных данных (D2, D3 или D4) с целью формирования соответствующих декодированных выходных данных (D5), включающий схему обработки данных для обработки одной или более частей кодированных данных (D2, D3 или D4), при этом схема обработки данных выполнена с возможностью применения одного из видов разностного и/или суммирующего декодирования к одной или более соответствующим кодированным последовательностям упомянутых одной или более частей, при этом одну или более кодированные последовательности подвергают операции циклического перехода относительно максимального значения и/или циклического перехода относительно минимального значения с целью формирования декодированных выходных данных (D5).

Опционально, декодер выполнен с возможностью декодирования кодированных данных (D2, D3 или D4), включающих одно или более 1-битных значений, при этом декодер выполнен с возможностью декодирования кодированных данных (D2, D3 или D4) побитно.

Опционально, декодер сконфигурирован таким образом, что схема обработки данных выполнена с возможностью приема одного или более значений для применения к одной или более кодированным последовательностям с целью использования при формировании декодированных выходных данных (D5). Также, опционально, принятые значения являются максимальными значениями, минимальными значениями и/или значениями смещения. Также, опционально, декодер выполнен с возможностью функционирования так, что одно или более значений смещения имеют значение «0».

Опционально, декодер выполнен с возможностью функционирования в случае, когда одна или более кодированных последовательностей представляют изменения в последовательных значениях, закодированных в кодированных данных (D2, D3 или D4).

Опционально, декодер выполнен с возможностью применения значения циклического перехода (wrapValue), при этом значение циклического перехода равно максимальному значению (highValue) минус минимальное значение (lowValue) плюс 1. Это значение циклического перехода используют, предпочтительно, при формировании кодированных выходных данных (D5), чтобы исключить появление таких значений, которые были бы меньше минимального значения (lowValue) (т.е. часто, отрицательным значением) или больше максимального значения (highValue).

Опционально, декодер сконфигурирован таким образом, что схема обработки данных

выполнена с возможностью применения к данным, обрабатываемым при ее помощи, операции, обратной по меньшей мере одному из следующего: кодирование длин серий (RLE), раздельное RLE-кодирование, модификатор энтропии (EM), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование.

Опционально, схема обработки данных выполнена с возможностью предположения задаваемого по умолчанию значения для первого значения предсказания в последовательности данных, декодируемых с ее помощью. Также, опционально, заданное по умолчанию значение предсказания имеет значение «0». Однако значение предсказания может также представлять собой минимальное значение (lowValue), максимальное значение (highValue), (максимальное значение (highValue) плюс минимальное значение (lowValue) плюс 1) разделить на 2 и т.п., как это упоминалось выше.

Опционально, в декодере, схема обработки данных реализована с использованием вычислительной аппаратуры, выполненной с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

В соответствии с четвертым аспектом настоящего изобретения предложен способ использования декодера для декодирования кодированных данных (D2, D3 или D4) с целью формирования соответствующих декодированных выходных данных (D5), включающий:

использование схемы обработки данных для обработки одной или более частей кодированных данных (D2, D3 или D4), при этом схема обработки данных выполнена с возможностью применения одного из видов разностного и/или суммирующего декодирования к одной или более соответствующим кодированным последовательностям упомянутых одной или более частей, при этом одну или более кодированных последовательностей подвергают операции циклического перехода относительно максимального значения и/или циклического перехода относительно минимального значения с целью формирования декодированных выходных данных (D5).

Опционально, способ включает декодирование кодированных данных (D2, D3 или D4), включающих одно или более 1-битных значений, и использование декодера для декодирования кодированных данных (D2, D3 или D4) побитно. Также, опционально, в способе, одно или более значений смещения имеют значение «0».

Опционально, способ включает использование схемы обработки данных для приема значений входных данных (D1) и/или одной или более соответствующих кодированных последовательностей (D2, D3 или D4) с целью применения их к одной или более соответствующим кодированным последовательностям для использования при формировании кодированных выходных данных (D5).

Опционально, в способе одна или более соответствующих кодированных последовательностей представляют изменения в последовательных значениях, закодированных в кодированных данных (D2, D3 или D4).

Опционально, способ включает использование циклического перехода относительно максимального значения или циклического перехода относительно минимального значения с целью исключения возникновения отрицательных разностей или слишком больших значений при формировании декодированных выходных данных (D5).

Опционально, в способе схема обработки данных выполнена с возможностью применения к данным, обрабатываемым при ее помощи, обработки, обратной по

меньшей мере одному из следующего: кодирование длин серий (RLE), раздельное RLE-кодирование, модификатор энтропии (EM), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование.

Опционально, схема обработки данных выполнена с возможностью предположения задаваемого по умолчанию значения для первого значения предсказания в последовательности данных, декодируемых с ее помощью. Также, опционально, в способе заданное по умолчанию значение предсказания имеет значение «0».

Также, опционально, в способе схему обработки данных реализуют с использованием вычислительной аппаратуры, выполненной с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

В соответствии с пятым аспектом настоящего изобретения предложен способ использования декодера для декодирования кодированных данных (D2, D3 или D4) с целью формирования соответствующих декодированных выходных данных (D5), включающий:

(a) использование схемы обработки данных для обработки кодированных данных (D2, D3 или D4) с целью применения декодирования к одной или более частям кодированных данных (D2, D3 или D4), с учетом того, что кодированные данные (D2, D3 или D4) содержат по меньшей мере одну кодированную последовательность, представляющую изменения в последовательных значениях смещенных данных, и используют циклический переход относительно максимального значения или циклический переход относительно минимального значения; и

(b) использование схемы обработки данных для формирования соответствующих обработанных данных, а также для смещения упомянутых одной или более частей с использованием по меньшей мере одного значения предварительного смещения и/или пост-смещения с целью формирования декодированных выходных данных (D5).

Что касается описанных выше аспектов, в отношении значений смещения, опционально, в декодере, схема обработки данных выполнена с возможностью приема диапазона значений кодированных данных (D2, D3 или D4) и для вычисления на его основе по меньшей мере одного значения предварительного смещения и/или пост-смещения. Значение пост-смещения может использоваться для формирования смещенных данных перед применением обратного оператора, например, оператора обратного O-дельта-кодирования, а значение предварительного смещения используют для смещения данных после применения обратного оператора. Это по меньшей мере одно значение смещения, опционально, комбинируют с обработанными данными с целью получения декодированных выходных данных (D5).

Смещение, опционально, поддерживается в декодере. Смещение используют тогда, когда оно использовано также и в кодере. Циклический переход в пределах параметра (wrapValue), для которого используется диапазон (lowValue, highValue), обратный оператор (суммирующий либо дифференциальный) и обратное устройство предсказания (входное значение на основе выходного значения) являются важными элементами декодера.

Опционально, в декодере схема обработки данных выполнена с возможностью применения к данным, обрабатываемым при ее помощи, обработки, обратной по меньшей мере одному из следующего: кодирование длин серий (RLE), раздельное RLE-кодирование, модификатор энтропии (EM), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование. Обработку выполняют с целью формирования данных (D4) на основе данных (D3).

В декодере, схема обработки данных выполнена с возможностью применения циклического перехода в пределах параметра (wrapValue), для которого использован диапазон (lowValue, highValue), при реализации в виде обратного декодирования, например, обратного O-дельта-декодирования.

В соответствии с шестым аспектом настоящего изобретения предложен кодек, включающий по меньшей мере один кодер по первому аспекту настоящего изобретения, для кодирования входных данных (D1) с целью формирования соответствующих кодированных данных (D2 или D3), а также по меньшей мере один декодер по третьему аспекту настоящего изобретения, для декодирования кодированных данных (D2, D3 или D4) с целью формирования соответствующих декодированных данных (D5).

В соответствии с седьмым аспектом настоящего изобретения предложен программный продукт, записанный на машиночитаемом носителе для хранения данных и являющийся исполняемым на вычислительном оборудовании с целью выполнения способа кодирования данных по второму аспекту настоящего изобретения.

В соответствии с восьмым аспектом настоящего изобретения предложен программный продукт, записанный на машиночитаемом носителе для хранения данных и являющийся исполняемым на вычислительном оборудовании с целью исполнения способа декодирования данных по четвертому аспекту настоящего изобретения.

Нужно понимать, что отличительные признаки настоящего изобретения могут комбинироваться произвольным образом в пределах объема настоящего изобретения, заданного приложенной формулой изобретения.

Описание чертежей

Далее, исключительно в качестве примера и со ссылками на приложенные чертежи, будут описаны примеры осуществления настоящего изобретения, где:

Фиг. 1 представляет собой иллюстрацию кодека, включающего кодер и декодер, реализованных для функционирования согласно настоящему изобретению;

Фиг. 2 представляет собой иллюстрацию шагов способа кодирования данных, выполняемого в кодере, показанном на фиг. 1; и

Фиг. 3 представляет собой иллюстрацию шагов способа декодирования данных, выполняемого в декодере, показанном на фиг. 1.

На приложенных чертежах числа, выделенные подчеркиванием, используются для обозначения элементов, над которыми находится подчеркнутое число, или с которыми подчеркнутое число является смежным. Неподчеркнутые числовые обозначения относятся к объектам, указанным линией, которая соединяет неподчеркнутое число и объект. Если число не выделено подчеркиванием и сопровождается связанной с ним стрелкой, это неподчеркнутое число используется для обозначения общего элемента, на который указывает стрелка.

Описание вариантов осуществления изобретения

При описании вариантов осуществления настоящего изобретения будут использованы следующие сокращения и определения, приведенные в таблице 2:

Таблица 2: Сокращения и определения

Сокращение	Описание
ADC	Аналогово-цифровой преобразователь (analog-to-digital converter)
Кодек	Кодер и соответствующий декодер для цифровых данных
DAC	Цифро-аналоговый преобразователь (digital-to-analog converter)
DB	База данных (database) в памяти с произвольным доступом (Random Access Memory, RAM) или в памяти в режиме «только для чтения» (Read Only Memory, ROM)
DC	постоянная составляющая (DC) исходного изображения, то есть, среднее значение изображения, а именно, значение, соответствующее средней яркости и представляющее собой низшую составляющую пространственной частоты изображения
RLE	Кодирование длин серий
ROI	Анализируемая область (Region of interest)
ROM	Память «только для чтения»
VLC	Кодирование переменной длины

В целом, на примере фиг. 1, настоящее изобретение относится к кодеру 10 и связанному с ним способу функционирования; предпочтительно, кодер 10 реализован в виде прямого O-дельта-кодера. При этом настоящее изобретение относится также к соответствующему декодеру 20; предпочтительно, декодер 20 реализован в виде обратного O-дельта-декодера. В вариантах осуществления настоящего изобретения, предпочтительно, применяется оператор прямого O-дельта-кодирования, который является вариантом упомянутого выше существующего способа дельта-кодирования, оптимизированным для побитного кодирования, а также вариантом, оптимизированным по диапазону в случае других данных. O-дельта-кодирование используют в вычислительном оборудовании или специальном цифровом оборудовании, где применяются информационные слова переменной длины, например, 8/16/32/64 бит, и/или где применяется кодирование переменной длины для 8/16/32/64-битных элементов данных, исходное значение которых выражено в диапазоне от 1 до 64 бит, и соответствующее кодированное значение формируют с использованием от 1 до 64 бит. Очевидно, кодер 10 и декодер 20 являются, в любом случае, осведомленными о типе числовых значений, присутствующих в данных D1, например, исходных данных, и соответственно их определение или передача не будут рассматриваться здесь более подробно. Просто предполагается, что известен диапазон чисел (от MIN до MAX), и что данные D1 подходят для применения.

Существующие методы дельта-кодирования расширяют диапазон значений от исходного (MIN, MAX) до результирующего (MIN-MAX, MAX-MIN). Это означает, что они также создают отрицательные значения, даже если исходные данные содержат только положительные значения. Оператор O-дельта-кодирования, соответствующий настоящему изобретению, никогда не создает значений, не лежащих в диапазоне

соответствующих исходных значений, и не расширяет используемого диапазона данных, и следовательно, эффективно применяется, например, при понижении энтропии и соответствующем сжатии данных. Рассмотрим пример, в котором при помощи существующих способов дельта-кодирования обрабатывают потоки 5-битных данных, а именно, данных, лежащих в диапазоне значений от 0 до 31: значения данных, формируемых такими способами дельта-кодирования будут лежать в диапазоне от -31 до +31, а именно, будут присутствовать 63 значения, которые, соответственно, могут быть представлены с использованием 6 бит (т.е. знакового бита +5 бит); в отличие от них, оператор прямого O-дельта-кодирования при работе с упомянутыми выше потоками 5-битных данных будет формировать значения, также лежащие в диапазоне от 0 до 31. При этом существующие способы дельта-кодирования не допускают рекурсивной реализации, тогда как оператор прямого или обратного O-дельта-кодирования в соответствии с настоящим изобретением может быть реализован рекурсивно и при этом также сохраняет используемый диапазон значений. Диапазон значений не обязательно должен иметь точность до одного бита, например, если значения от 0 до 31 представляемы 5 битами, оператор O-дельта-кодирования может использовать любой диапазон значений, к примеру, диапазон значений от 0 до 25, и при этом также функционировать корректным образом.

В принципе, способы O-дельта-кодирования, описанные в настоящем изобретении, всегда способны функционировать на базе имеющегося диапазона данных, примеры чего будут рассмотрены ниже. Способы O-дельта-кодирования могут быть усовершенствованы за счет предоставления информации, указывающей наименьшее встречающееся в данных значение ("lowValue") и наибольшее встречающееся в данных значение ("highValue"). Нужно отметить, что lowValue>=MIN, a highValue<=MAX, и что эти значения являются опциональными.

Ниже описаны два примера операторов прямого и обратного O-дельта-кодирования в соответствии с настоящим изобретением. Первый пример операторов прямого и обратного O-дельта-кодирования является эффективным и относительно простым в реализации, например, в электронном и/или вычислительном оборудовании, выполненном с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

При реализации операторов прямого или обратного O-дельта-кодирования в соответствии с настоящим изобретением, предпочтительно, все значения данных в исходной последовательности являются положительными, и наименьшее значение равно 0. Опционально, может использоваться некоторое значение смещения, например, значение предварительного смещения или значение пост-смещения, для смещения значения данных таким образом, чтобы все они были положительными и наименьшим значением был «0». Оператор O-дельта-кодирования в соответствии с настоящим изобретением допускает непосредственное применение с любыми типами данных; как правило, он позволяет получить сжатие данных, а именно, снизить требуемую скорость передачи данных, поскольку при добавлении значения смещения ко всем значениям или при вычитании значения смещения изо всех значений диапазон значений данных может быть задан с использованием меньшего количества битов. Например, если исходные значения данных, перед применением оператора прямого или обратного O-дельта-кодирования лежат в диапазоне от -11 до +18, этот диапазон может быть преобразован в диапазон от 0 до 29 с помощью значения смещения, равного 11, при этом преобразованный диапазон, соответственно, может быть представлен 5 битами. Если подобное значение предварительного смещения или значение пост-смещения не

применяют, исходные данные требуют для своего описания по меньшей мере 6 битов, а на практике, для удобства, часто применяют полный 8-битный байт со знаком.

Аналогичная оптимизация диапазона данных также возможна при использовании обобщенного оператора прямого или обратного O-дельта-кодирования. То есть, если с помощью оператора прямого или обратного O-дельта-кодирования, или какого-либо иного способа, формируют значения данных, которые могут быть представлены, с использованием значения смещения, меньшим количеством битов, чем полный диапазон значений, то такая оптимизация диапазона может выполняться на любом из этапов способа O-дельта-кодирования. Когда используют значение смещения, будь оно отрицательным или положительным, его необходимо передавать из кодера 10 в декодер 20, как это показано ниже на примере фиг. 1, фиг. 2 и фиг. 3.

Оператор прямого O-дельта-кодирования может иметь 1-битную реализацию, например, для побитного кодирования исходных данных D1; в случае 1-битной реализации способ 1 и способ 3, в соответствии с последующим более подробным описанием, формируют значение «0», когда в исходных данных, проиллюстрированных на фиг. 1, отсутствуют изменения битового значения, и значение «1», когда происходит изменение битового значения. Предсказание для первого бита в исходных данных, опционально, имеет значение «0», и соответственно, значение первого бита исходных данных D1 остается неизменным. Альтернативно и опционально возможно применение значения предсказания для первого бита в исходных данных, имеющего значение «1», но такое решение не дает никаких преимуществ при кодировании; по этой причине, если, опционально, предсказание для 1-битных данных всегда подразумевается равным «0», отсутствуют необходимость сообщения о выборе, то есть заранее заданное значение «0» применяют для кодера 10 и декодера 20, что снимает необходимость передачи информации об этом значении предсказания, и следовательно, повышает эффективность сжатия данных.

Ниже описан один из примеров прямого O-дельта-кодирования в соответствии с настоящим изобретением. В уравнении 1 представлен пример исходной последовательности битов, а именно, двадцать семь битов, включающих 17 единиц («1») и двадцать нулей («0»):

$$0\,1\,0\,1\,0\,1\,1\,0\,0\,1\,0\,0\,0\,1\,0\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,1$$

Ур. 1

ее энтропию E вычисляют на основе уравнения 2 (ур. 2):

$$E = 17 * \log_{10}\left(\frac{37}{17}\right) + 20 * \left(\frac{37}{20}\right) = 11.08523$$

Ур. 2

Количество битов, Min_bit, необходимое для кодирования энтропии E в уравнении 2 (ур. 2), может быть вычислено на основе теоремы Шеннона о кодировании источника, описанной в упомянутых выше документах D7 и D8, как это показано в уравнении 3 (ур. 3):

$$Min\_bits = \frac{E}{\log_{10}(2)} = 36.82 \text{ бит}$$

Ур. 3

При выполнении обработки исходной битовой последовательности оператором прямого О-дельта-кодирования в соответствии с предшествующим описанием, а именно способом 1 и способом 3, получают следующую последовательность битов, включающую тридцать семь битов, в которой имеются тринадцать единиц («1») и двадцать четыре нуля («0»):

0 1 1 1 1 1 0 1 0 1 1 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0

Ур. 4

ее энтропию E вычисляют на основе уравнения 5 (ур. 5):

$$E = 13 * \log_{10}\left(\frac{37}{13}\right) + 20 * \left(\frac{37}{24}\right) = 10.41713$$

Ур. 5

Что может быть выражено минимальным количеством бит, Min_bits, полученным согласно уравнению 6 (ур. 6):

$$Min\_bits = \frac{E}{\log_{10}(2)} = 34.60 \text{ бит}$$

Ур. 6

Битовую последовательность из уравнения 4 (ур. 4), предпочтительно подвергают дополнительному кодированию с целью сжатия данных, например, с использованием по меньшей мере одного из следующего: кодирование длин серий (RLE), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование, модификатор энтропии или SRLE-кодирование.

Оператор О-дельта-кодирования позволяет уменьшить количество битов, необходимых для представления исходных данных D1 при применении соответствующего способа энтропийного кодирования, например, RLE- или SRLE-кодирование выполняют над данными после применения оператора, к примеру, над данными из уравнения 4 (ур. 4), вместо исходных данных из уравнения 1 (ур. 1); 1-битный вариант оператора прямого О-дельта-кодирования, а именно способ 1 и способ 3, формирует единицы («1»), когда в исходной битовой последовательности из уравнения 1 (ур.1) присутствует много изменений, и формирует нули («0»), когда в исходной последовательности по уравнению 1 (ур. 1) имеется длинный поток одинаковых битов.

Обратный вариант оператора О-дельта-кодирования, является буквально обратным способу 1 и способу 3, т.е. в нем битовые значения «0» заменяют на значения «1», а значения «1» заменяют на значения «0», когда в кодированном потоке данных, т.е. в данных D2, присутствует значение «1», и не изменяют битовое значение, когда в кодированном потоке данных D2 присутствует значение «0». Когда такой оператор О-дельта-кодирования применяют к битовому потоку, полученному в результате применения оператора прямого О-дельта-кодирования, то в качестве декодированных данных D5 получают исходных поток данных D1; однако, как упоминалось ранее,

предпочтительно, применяют дополнительное кодирование, такое как VLC-кодирование или кодирование Хаффмана, что тоже необходимо учитывать; это означает, что данные D3 формируют на основе данных D2 с использованием прямой операции энтропийного кодирования, а данные D4 формируют на основе данных D3 с использованием обратной операции энтропийного кодирования.

Предпочтительно, исходный поток данных D1, перед применением к нему кодирования, разбивают на два или более фрагмента. Такое разбиение дает возможность получить большую оптимизацию при кодировании исходного потока данных D1. Например, такое разбиение является предпочтительным, потому что изменяющиеся последовательности в данных D1 дают больше единиц («1») при прямом O-дельта-кодировании, то есть при применении способа 1 и способа 3, тогда как ровные, неизменяющиеся последовательности, то есть «ровные» последовательности, дают больше нулей «0», что, например, является более выгодным для последующего VLR-кодирования или кодирования Хаффмана, поэтому энтропия E может быть снижена для всего битового потока, образующего данные D1, за счет разбиения его на множество фрагментов, которые могут кодироваться отдельно в соответствии с предыдущим описанием.

Далее будет описан один из примеров прямого O-дельта-кодирования согласно настоящему изобретению, в котором применяют множество отдельно кодируемых фрагментов. Первый фрагмент, включающий последовательность исходных одиночных битов, включает в общей сложности шестнадцать битов, а именно, семь единиц («1») и девять нулей («0»), в соответствии с уравнением 7 (ур. 7):

0 1 0 1 0 1 1 0 0 1 0 0 0 1 0 1

Ур. 7

где H(X)=4,7621, а B=15,82, H - энтропия, а B - это Max_bit. Когда последовательность исходных битов из уравнения 7 (ур. 7) подвергают обработке прямым оператором O-дельта-кодирования, получают последовательность соответствующих преобразованных бит, в соответствии с уравнением 8 (ур. 8):

0 1 1 1 1 1 0 1 0 1 1 0 0 1 1

Ур. 8

где H(X)=4,3158, а B=14,34.

Второй фрагмент, включающий последовательность исходных одиночных битов, соответствует уравнению 9 (ур. 9):

0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 **Ур. 9**

где H(X)=6,3113, а B=20,97. Когда последовательность исходных битов из уравнения 9 (ур. 9) подвергают обработке прямым оператором O-дельта-кодирования, получают последовательность соответствующих преобразованных битов, в соответствии с уравнением 10 (ур. 10):

0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0

Ур. 10

где H(X)=1,7460, а B=5,80. В данных примерах, как упоминалось ранее, H(X) представляет собой энтропию E, а B представляет собой минимальное количество битов, необходимых для кодирования.

Наилучшее сжатие в данном примере на основе уравнения 7 (ур. 7) и уравнения 10

(ур.) получают, когда оба фрагмента отдельно подвергают обработке прямым оператором О-дельта-кодирования (а именно, кодирование в 14,34 битах + 5,80 битах = 20,14 бит в общей сложности); это меньше, чем 36,82 бита, требуемых изначально, а именно, меньше 34,60 бит, требуемых для битов после применения оператора прямого О-дельта-кодирования, или исходного количества битов, требуемого после разбиения (= 15,82 бит + 20,97 бит = 36,79 бит). Предпочтительно, разбиение исходного битового потока в данных D1 на фрагменты выполняют автоматически при помощи анализа энтропии E исходных данных D1 и соответствующей энтропии H модифицированных данных, то есть данных, входящих в состав данных D2, по частям.

Сжатие данных, опционально, реализуют «грубо», просто разбивая фрагменты данных D1 на новые кодируемые фрагменты, если в данных D1, имеются множество фрагментов с длинными сериями, при условии, что на протяжении последовательности есть достаточна большая область данных, в которой значения битов быстро изменяются. Опционально, некоторые из фрагментов данных D1 кодируют без применения оператора О-дельта-кодирования, например, если имеется длинная серия одинаковых битов с относительно малым количеством отдельных отличающихся битов между ними; в таком случае оператор прямого О-дельта-кодирования не обладает значительными преимуществами для сжатия данных.

Разбиение данных D1 на менее крупные фрагменты имеет недостаток, заключающийся в необходимости формирования дополнительной служебной информации, которая увеличивает объем кодированных данных D2. Такая служебная информация включает, например, информацию, указывающую на количество битов или байтов данных, связанных с каждым новым фрагментом. Однако в любом случае передача по меньшей мере некоторого объема служебных данных является необходимой, и следовательно, при разбиении исходных данных на два фрагмента будет присутствовать лишь небольшой дополнительный объем служебных данных.

Для получения кодированного битового потока, который позднее может быть декодирован, предпочтительно, после применения оператора прямого О-дельта-кодирования реализуют энтропийное кодирование, например, VLC-кодирование, кодирование Хаффмана, арифметическое кодирование, интервальное кодирование, RLE-кодирование, S RLE-кодирование, модификатор энтропии и т.п. По сравнению с фактическим кодированием данных, выполнение оптимизационных вычислений на основе вычисленной энтропии E и значений оценки минимальных битов является более простым и вычислительно эффективным. Такой порядок выполнения позволяет получить значительное повышение скорости, и часто также дает оптимальный результат в отношении сжатия кодированных данных D2. Альтернативно, оптимизация энтропии может выполняться таким образом, что сначала кодируют данные исходного бита, алфавитного символа, числа, байта или слова из данных D1 с помощью некоторого другого способа, в результате чего получают битовый поток, оптимизированный по энтропии, и после этого применяют оператор прямого О-дельта-кодирования для модификации энтропийно-оптимизированного битового потока, в результате чего получают соответствующие кодированные данные, а именно, данные D2. При этом данные, полученные применением оператора О-дельта-кодирования, могут также кодироваться с использованием других способов кодирования, с получением данных D3 на основе данных D2.

В обобщенном операторе прямого О-дельта-кодирования применяют параметр, который описывает диапазон значений, используемых в данных D1, то есть значение или число битов, которые необходимы для представления этих значений. При этом

оператор O-дельта-кодирования применяют в способе, который позволяет использовать положительные и отрицательные значения смещения, или, другими словами, положительные и отрицательные значения «пьедестала». Например, если данные D1 представлены семью битами, то есть имеют возможные значения от 0 до 127, но содержат только значения в диапазоне от 60 до 115, то, когда к данным D1 прикладывают значение смещения, равное -60, в результате получают смещенные данные, имеющие значения в диапазоне от 0 до 55, которые при этом могут быть представлены как значения, содержащие только шесть битов, то есть таким образом может быть достигнута определенная степень сжатия данных. Таким образом, обобщенный оператор прямого O-дельта-кодирования дает улучшенные результаты, когда в данных D1 присутствует полный диапазон значений данных, то есть представляемых семью битами и для удобства представленных 8-битными байтами.

В соответствии с настоящим изобретением значения прямого O-дельта-кодирования, а именно, способа 1, могут быть вычислены с помощью процедуры, описанной примером фрагмента программного кода, приведенного ниже, для данных, которые имеют только положительные значения (lowValue=MIN=0 и highValue=MAX=127, wrapValue=127-0+1=128):

```
wrapValue = power(2, bits) = power(2, 7) = 128
prediction Value = (lowValue + highValue + 1) div 2 = (wrapValue + 1) div 2 + lowValue = 64
for all pixels
begin
        if(originalValue >= predictionValue) then
                ODeltaValue = originalValue – predictionValue
        else
                ODeltaValue = wrapValue + originalValue – predictionValue
predictionValue = originalValue
End
```

Для более детального описания упомянутого выше оператора O-дельта-кодирования ниже будет приведен еще один пример. Исходная последовательность значения соответствует уравнению 11 (ур. 11):

65, 80, 126, 1, 62, 45, 89, 54, 66 Ур. 11

Соответствующие значения дельта-кодирования приведены в уравнении 12 (ур. 12):

65, 15, 46, -125, 61, -17, 44, -35, 12

Ур. 12

Соответствующие значения O-дельта-кодирования приведены в уравнении 13 (ур. 13):

1, 15, 46, 3, 61, 111, 44, 93, 12 Ур. 13

где был применен циклический переход в пределах параметра wrapValue.

Оператор обратного O-дельта-кодирования, а именно, способ 1, может применяться

для формирования значений обратного О-дельта-кодирования, например, в соответствии с реализацией при помощи следующего примера программного кода:

```
wrapValue = power(2, bits) = power(2, 7) = 128
predictionValue = (wrapValue + 1) div 2 + lowValue = 64
for all pixels
begin
        ODelta Value = originalValue + predictionValue
        if (ODeltaValue >= wrapValue) then
                ODelta Value = О-дельта-кодированиеValue – wrapValue
        predictionValue = ODeltaValue
end
```

При исполнении этого программного кода и применении его к уравнению 13 (ур. 13), он формирует значения, приведенные в уравнении 14 (ур. 14):

65, 80, 126, 1, 62, 45, 89, 54, 66 Ур. 14

В данном примере в качестве значения power-of-two (степень двойки) используется wrapValue. Однако это не является обязательным, и wrapValue может иметь любое значение, большее, чем наибольшее значение данных, или значение, больше, чем используемый диапазон, если имеются также отрицательные значения, или диапазон в исходной последовательности данных модифицируют с помощью предварительного смещения. Это будет более подробно проиллюстрировано еще на одном примере, приведенном ниже.

Итак, подытоживая предшествующее описание на примере фиг. 1, настоящее изобретение относится к кодеру 10 и декодеру 20. Опционально, кодер 10 и декодер 20 применяют совместно в виде кодека, обозначенного в целом позицией 30. Кодер 10 выполнен с возможностью приема исходных входных данных D1, которые кодируют, например, с использованием способа прямого О-дельта-кодирования, и получают соответствующие кодированные данные D2 или D3. Кодированные данные D2 или D3, опционально, передают по сети 40 связи, или сохраняют на носителе 50 для хранения данных, например, таком носителе данных, как оптический диск, память «только для чтения» (ROM) или аналогичном носителе. Декодер 20 выполнен с возможностью приема кодированных данных D2 или D3, например передаваемых в виде потока по сети 40 связи или предоставленных на носителе 50 для хранения данных, и с возможностью применения обратного способа, например, способа обратного О-дельта-кодирования, в результате чего получают соответствующие декодированные данные D5, например, по существу аналогичные исходным данным D1. Кодер 10 и декодер 20 реализуют, предпочтительно, с использованием цифровой аппаратуры, например, вычислительной аппаратуры, которая выполнена с возможностью исполнения одного или более программных продуктов, например, кодов, приведенных в качестве примеров осуществления настоящего изобретения в настоящем описании. Альтернативно, кодер 10 и/или декодер 20 реализуют с использованием специальной цифровой аппаратуры.

В способе О-дельта-кодирования, исполняемом в кодере 10, применяют шаги в соответствии с иллюстрацией фиг. 2. На первом шаге 100, являющемся опциональным,

обрабатывают входные данные D1 для определения диапазона значений элементов этих данных. На втором шаге 110, являющемся опциональным, на основе диапазона значений вычисляют смещение, необходимое для смещения элементов данных в положительные значения, в результате чего получают соответствующий набор смещенных элементов. На третьем шаге 120 элементы, опционально смещенные на втором шаге 110, подвергают прямому O-дельта-кодированию, в результате чего получают соответствующие O-дельта-кодированные значения. На четвертом шаге 130 O-дельта-кодированные значения и опциональное значение смещения, минимальное значение (lowValue) и/или максимальное значение (highValue) отдельно кодируют, например, с использованием кодирования длин серий (RLE), интервального кодирования или кодирования Хаффмана, в результате чего на основе данных D2 формируют данные D3. Значение смещения, минимальное значение (lowValue) и/или максимальное значение (highValue) не всегда могут быть сжаты, что требуется для их передачи, с использованием подходящего количества битов, из кодера 10 в декодер 20. При этом значение смещения, минимальное значение (lowValue) и/или максимальное значение (highValue) по отношению к прямому оператору O-дельта-кодирования являются опциональными элементами; к примеру, в некоторых ситуациях значение смещения имеет значение «0», lowValue имеет значение MIN, а highValue имеет значение MAX, то есть смещение не применяется, и используется полный диапазон. В частности, когда оператор прямого O-дельта-кодирования реализуют для 1-битных данных, то есть для побитного кодирования, он совсем не требует значения смещения, и тогда шаги 100 и 110 во всех случаях опускают. Если на шаге 110 используют также значение смещения, вместе с ним должно быть обновлено значение диапазона, отражающее наибольшее и наименьшее значения. Количество различных значений, то есть wrapValue, также должно быть известно декодеру 20, или, если это не так, кодер 10 должен передавать его в декодер 20 в составе сжатых данных. Опционально в кодере и декодере используют заданное по умолчанию значение wrapValue (= highValue - lowValue + 1). Опционально, кодер 10 и/или декодер 20 функционируют рекурсивно, например, для поиска оптимального варианта разбиения входных данных D1, для их кодирования, на фрагменты, что позволяет получить оптимальное сжатие данных D1, в результате чего формируют кодированные данные D2.

В способе обратного O-дельта-кодирования, исполняемом в декодере 20, применяют шаги в соответствии с иллюстрацией фиг. 3. На первом шаге 200 данные D2/D3 или D4 подвергают кодированию, которое обратно применяемому на описанном выше шаге 130, и получают O-дельта-декодированные данные, при этом O-дельта-декодированные данные содержат O-дельта-кодированные значения и, опционально, имеют отдельное значение смещения. На втором шаге 210 O-дельта-кодированные значения декодируют и формируют последовательность элементов данных. На третьем шаге 220 последовательность элементов данных смещают с использованием опционального значения предварительного смещения и формируют декодированные данные D5; в определенных ситуациях такое смещение задано равным «0», то есть по существу смещение не применяется. Также способ может быть выполнен без необходимости применения значения смещения, например, при осуществлении 1-битного кодирования, то есть побитного кодирования, в этом случае шаг 220 может быть опущен. При этом декодеру 20, чтобы он был способен корректно декодировать принятые в нем элементы данных, должно также быть известно значение wrapValue.

За счет применения смещения и получения только положительных значений, может быть достигнуто более эффективное сжатие в данных D2 или D3. Если все значения

данных уже являются положительными, то добавлять какое-либо значение смещения не является необходимым. Очевидно, отрицательные значения смещения, опционально, могут быть использованы для сужения используемого диапазона, как это показано в следующем примере, однако не являются обязательными.

Способы, показанные на фиг. 2 и фиг. 3, могут быть, опционально, дополнительно оптимизированы за счет использования только реально имеющихся значений, к которым применяют О-дельта-кодирования. Такая оптимизация требует, чтобы используемые значения были известны. Например, в одном из приведенных выше примеров, в исходном наборе D1 данных присутствуют только значения от 1 (= исходный минимум) до 126 (исходный максимум). В этом случае значение смещения равно 1 (-> lowValue = исходный минимум - смещение = 1-1=0, и highValue = исходный максимум - смещение = 126-1=125) После вычитания из исходных данных D1 значения предварительного смещения в результате получают следующие значения, приведенные в уравнении 15 (ур. 15):

64, 79, 125, 0, 61, 44, 88, 53, 65 **Ур. 15**

Из уравнения 15 (ур. 15) определяют, что максимальное значение равно 125 (highValue = исходный максимум - смещение = 126-1=125), так что «количество» (= максимальное значение разности = highValue - lowValue) может быть в этом случае равно 125, или наименьшее значение wrapValue может быть равно 126 (= «количество» + 1 = highValue - lowValue + 1). Далее необходимо сохранить и/или передать эти значения, и тогда предыдущий пример может быть модифицирован, при помощи изменения значений процедуры, следующим образом:

wrapValue = 126 ("0" to "125" => 126 different values)

prediction Value = (lowValue + highValue + 1) div 2 = (wrapValue + 1) div 2 + lowValue = 63

Соответствующие значения после применения оператора О-дельта-кодирования приведены в уравнении 16 (ур. 16):

1, 15, 46, 1, 61, 109, 44, 91, 12 **Ур. 16**

Нужно понимать, что «все отрицательные значения разности» уменьшены в два раза (то есть изменение диапазона = 128-126). Аналогично, в декодере 20 значения процедуры должны быть изменены следующим образом:

wrapValue = 126

predictionValue = (wrapValue + 1) div 2 + lowValue = 63

Соответствующие значения обратного О-дельта-кодирования приведены в уравнении 17 (ур. 17):

64, 79, 125, 0, 61, 44, 88, 53, 65 **Ур. 17**

Когда к уравнению 17 (ур. 17) добавляют предварительное смещение, получают приведенный в уравнении 18 (ур. 18) результат, соответствующий исходным данным из уравнения 15 (ур. 15), а именно:

65, 80, 126, 1, 62, 45, 89, 54, 66 **Ур. 18**

В данном примере диапазон значений является почти полным, поэтому за счет применения оператора прямого О-дельта-кодирования со значением смещения и

максимальным значением (highValue) достигается лишь относительно небольшое преимущество. Однако при этом все же может быть достигнуто снижение энтропии E, что дает меньшее количество значений в частотной таблице или в таблице кодирования, когда они переданы корректно. Более значительная выгода может быть достигнута при меньшей степени использования диапазона.

Далее будет рассмотрен один из примеров осуществления применимого на практике 1-битного способа прямого и обратного O-дельта-кодирования, а именно, способа 1 и способа 3 кодирования данных. Пример будет продемонстрирован при помощи исполняемого компьютерного программного кода. В способе, то есть в способе 1 и способе 3, применяют описанные выше операторы прямого и обратного O-дельта-кодирования. Программный код выполнен с возможностью, при его исполнении в вычислительной аппаратуре, обработки данных из одного байтового буфера и помещения их в другой байтовый буфер. В программном коде функции GetBit (получить бит), SetBit (установить бит) и ClearBit (очистить бит) всегда обновляют значение HeaderBits (биты заголовка). Если следующий бит будет находиться в следующем байте, обновляют также значение Headerlndex (указатель на заголовок). Опционально, программный код может быть оптимизирован, в результате чего для источника и для назначения будет использоваться только один набор значений Headerlndex и HeaderBits, так что значения будут обновляться только при записи данного бита в буфер назначения.

```
procedure EncodeODelta1u(APtrSrc : PByte; ASrcDstBitLen : PCardinal; APtrDst : PByte)
var
 iSrcHeaderIndex, iSrcHeaderBits, iIndex,
 iDstHeaderIndex, iDstHeaderBits : Cardinal;
 bBit, bLastBit : Boolean;
begin
 // Reset offsets
 iSrcHeaderIndex := 0;
 iSrcHeaderBits := 0;
 iDstHeaderIndex := 0;
 iDstHeaderBits := 0;

 // Initialise delta value
 bLastBit := False;

 // Go through all bits
 for iIndex := 0 to ASrcDstBitLen^-1 do
 begin
  // Read bit
  bBit := GetBit(APtrSrc, @iSrcHeaderIndex, @iSrcHeaderBits);

  // Set destination bit if current source bit is different than previous source bit
  if (bBit <> bLastBit) then
  begin
   SetBit(APtrDst, @iDstHeaderIndex, @iDstHeaderBits);
   bLastBit := bBit;
  end
  else ClearBit(APtrDst, @iDstHeaderIndex, @iDstHeaderBits);
 end;
end;

function DecodeODelta1u(APtrSrc : PByte; ASrcDstBitLen : PCardinal; APtrDst : PByte) :
Boolean;
var
```

```
  iSrcHeaderIndex, iSrcHeaderBits, iIndex,
  iDstHeaderIndex, iDstHeaderBits : Cardinal;
  bBit, bLastBit : Boolean;
begin
  // Reset offsets
  iSrcHeaderIndex := 0;
  iSrcHeaderBits := 0;
  iDstHeaderIndex := 0;
  iDstHeaderBits := 0;

  // Initialise delta value
  bLastBit := False;

  // Go through all bits
  for iIndex := 0 to ASrcDstBitLen^-1 do
  begin
    // Read bit
    bBit := GetBit(APtrSrc, @iSrcHeaderIndex, @iSrcHeaderBits);

    // Change bit value if source bit is true
    if (bBit = True) then
    begin
      if (bLastBit = True) then
        bLastBit := False
      else bLastBit := True;
    end;

    // Set destination bit based on bit value (True or False)
    if (bLastBit) then
      SetBit(APtrDst, @iDstHeaderIndex, @iDstHeaderBits)
    else ClearBit(APtrDst, @iDstHeaderIndex, @iDstHeaderBits);
  end;
end;
```

Упомянутые выше прямые и обратные операторы О-дельта-кодирования, а именно,

способ 1 или способ 3, предпочтительно применяют для сжатия любых типов данных, имеющих цифровой формат, например, видеоданных, данных изображений, аудиоданных, графических данных, сейсмологических данных, медицинских данных, значений измерений, позиционных обозначений и масок. При этом с помощью оператора прямого O-дельта-кодирования могут быть сжаты также один или более аналоговых сигналов, если они сначала, перед сжатием, преобразованы в соответствующие цифровые данные, например, с использованием аналогово-цифровых преобразователей (ADC). Если используют оператор обратного O-дельта-кодирования, то после этой операции может применяться цифро-аналоговый преобразователь (DAC), если данные должны быть преобразованы в один или более аналоговых сигналов. Однако нужно понимать, что оператор прямого O-дельта-кодирования сам по себе, как правило, не эффективен для сжатия данных, но позволяет обеспечить эффективное сжатие данных при применении в комбинации с другими способами кодирования, например, кодирования переменной длины (VLC), арифметического кодирования, интервального кодирования, кодирования длин серий, SRLE-кодирования, модификатора энтропии и т.п. Такие способы кодирования применяют для данных 02 после применения оператора прямого O-дельта-кодирования в кодере 10. Кодированные данные D2 должны быть соответствующим образом декодированы в исходный вид до того, как результирующие данные передают в оператор обратного O-дельта-кодирования, реализованный в декодере 20. Оператор O-дельта-кодирования может также быть реализован в комбинации с другими типами модификаторов энтропии. В определенных ситуациях применение оператора прямого O-дельта-кодирования может давать повышение энтропии E, и при этом, предпочтительно, оператор прямого O-дельта-кодирования применяется алгоритмами сжатия данных избирательно, т.е. используется при кодировании данных только тогда, когда он положительно сказывается на степени сжатия данных, например, его применяют избирательно, исходя из характера сжимаемых данных, к примеру, избирательно применяют к избранным фрагментам входных данных D1, как это было описано выше.

Оператор прямого O-дельта-кодирования был создан, например, для применения в комбинации с блочным кодером, описанном в заявке на патент США 13/584005, которая полностью включена в настоящий документ путем ссылки, а оператор обратного O-дельта-кодирования был создан для применения в комбинации с блочным декодером, описанном в заявке на патент США 13/584047, которая полностью включена в настоящий документ путем ссылки. Опционально, оператор прямого O-дельта-кодирования и оператор обратного O-дельта-кодирования, предпочтительно, применяют в комбинации со способом многоуровневого кодирования, описанным в заявке на патент США 13/657382, которая полностью включена в настоящий документ путем ссылки. Предпочтительно, все типы 1-битных данных, к примеру, присутствующих в данных D1, которые включают двоичные состояния, подвергают обработке 1-битным вариантом оператора прямого O-дельта-кодирования, в результате чего получают соответствующие преобразованные данные, которые затем подвергают фактическому энтропийному кодированию с формированием кодированных данных D2 или D3. Опционально, в соответствии с предшествующим описанием, оператор прямого O-дельта-кодирования применяют избирательно в зависимости от характера исходных данных D1.

Опционально, возможно применение других способов изменения энтропии данных, до или после применения оператора прямого O-дельта-кодирования. К примеру, оператор прямого O-дельта-кодирования может также применяться непосредственно

для многобитных данных, с использованием обобщенного варианта оператора прямого О-дельта-кодирования. При этом описанный выше 1-битный вариант оператора прямого О-дельта-кодирования, предпочтительно, применяют для многобитных данных после того, как все используемые биты сначала помещают в специальную последовательность битов.

Когда в кодере 10, в комбинации с прямым оператором О-дельта-кодирования, применяют несколько способов сжатия данных, в декодере 20 соответствующие обратные операции выполняют в обратном порядке, например:

В кодере 10 применяют следующую последовательность способов:

[данные D1] => оператор прямого О-дельта-кодирования (способ 2)
=> VLC
=> EM => Арифметическое кодирование
[данные D3]

Ур. 19

В декодере 20 применяют следующую обратную последовательность способов:

[данные D3] => обратное арифметическое кодирование
=> обратный EM
=> обратное VLC
=> оператор обратного О-дельта-кодирования (способ 2)
=> [данные D5]

Ур. 20

где "VLC" - кодирование переменной длины, а "EM" - модификатор энтропии.

Оператор О-дельта-кодирования, описанный выше, обеспечивает обратимое кодирование без потерь. При этом оператор О-дельта-кодирования, опционально, может быть реализован специально для 1-битовых потоков данных, например, при выполнении побитного кодирования, а также и для других данных. Предпочтительно, любые данных могут быть обработаны с использованием обобщенного варианта оператора прямого О-дельта-кодирования. Предпочтительно, оператор прямого О-дельта-кодирования применяют в случае необходимости сжатия данных, а соответствующий оператор обратного О-дельта-кодирования применяют в случае необходимости декомпрессии сжатых данных. Опционально, если для сжатия применяют оператор обратного О-дельта-кодирования, оператор прямого О-дельта-кодирования и соответствующую ему обратную операцию применяют в обратном порядке; другими словами, сначала выполняют оператор обратного О-дельта-кодирования над исходным битовым потоком, и затем применяют оператор прямого О-дельта-кодирования для восстановления исходного битового потока. Один оператор О-дельта-кодирования повышает энтропию, а второй оператор О-дельта-кодирования уменьшает энтропию. Лишь в очень редких случаях оператор прямого О-дельта-кодирования совсем не меняет энтропию, и в этих случаях оператор обратного О-дельта-кодирования также не изменяет энтропию. Нужно отметить, что когда применяют операторы прямого и обратного О-дельта-кодирования, например, для способа 1, обратный порядок применения этих операторов соответствует нормальному порядку в способе 4. Подобное изменение порядка может выполняться также в способе 2 и способе 3.

В 1-битном, то есть предназначенном для побитного кодирования данных, варианте оператора прямого O-дельта-кодирования его применение начинается, предпочтительно, без предсказания, а именно, по умолчанию в нем предполагается предсказание исходного значения «O». В обобщенном варианте оператор прямого O-дельта-кодирования начинают применять с предсказания, которое представляет собой середину используемого диапазона данных; например, если для значений входных данных D1 используются 5 битов, то есть имеются тридцать два различных значения в диапазоне от 0 до 31, значение предсказания будет равно $^{32}/_{2} = 16$. Предпочтительно, в оператор O-дельта-кодирования должна быть предоставлена информация об используемом диапазоне данных для элементов данных, обрабатываемых с помощью оператора.

Варианты осуществления настоящего изобретения, описанные выше, обеспечивают возможность снижения энтропии E, присутствующей в данных D1, которые представлены в виде битов или числовых значений. Оператор прямого O-дельта-кодирования практически во всех случаях обеспечивает лучшее снижение энтропии, чем дельта-кодирование. Одинаковый результат достигается только в том случае, когда дельта-кодирование применяют в комбинации с байтовым циклическим переходом, и в разностном O-дельта-операторе с исходным предсказанием (способ 1) используют значения wrapValue = 256, lowValue = MIN = 0 и highValue = MAX = 255. Если используют другой способ прямого O-дельта-кодирования, или если во входных данных присутствует не полный диапазон, то оператор O-дельта-кодирования дает лучшие результаты путем передачи выбранного способа или lowValue, и/или highValue, то есть того, что автоматически изменяет wrapValue. Меньшая энтропия позволяет обеспечить более высокую степень сжатия данных. Более высокие степени сжатия данных позволяют применять меньшие объемы памяти для хранения данных, а также использовать меньшие полосы пропускания при передаче сжатых данных, что дает соответствующее снижение энергопотребления.

В предшествующем описании предполагалось, что в кодере 10 выполняется некоторая форма вычислений разности и суммы, а в декодере 20 выполняются соответствующие обратные вычисления. В кодере 10 допускается также применение других способов предсказания, и тогда соответствующее обратное предсказание будет применяться в декодере 20. Это означает, что фактически существуют по меньшей мере четыре различных способа прямого O-дельта-кодирования, а также по меньшей мере четыре соответствующих способа обратного O-дельта-кодирования. Ниже приведено подробное и полное описание этих способов. Опционально, вычисления могут выполняться рекурсивно, в результате чего в кодированных данных D2 (или D3) получают большую степень сжатия. При осуществлении подобных рекурсивных вычислений применяют изменение диапазона чисел как функцию выполненного количества рекурсивных вычислений. Например, в кодере 10, над данными D1 выполняют следующую последовательность вычислений, в результате чего формируют кодированные данные D2 (или D3):

[Данные D1] => (кодер) оператор прямого O-дельта-кодирования (способ 3)
=> (кодер) оператор прямого O-дельта-кодирования (способ 3)
(кодер) EM =>
=> (кодер) оператор прямого O-дельта-кодирования (способ 1)
(кодер) VLC [Данные D3] Ур. 21

а в декодере 20 выполняют соответствующие обратные операции:

[Данные D3] (декодер) VLC =>
=> (декодер) оператор обратного O-дельта-кодирования (способ 1)
(декодер) EM =>
=> (декодер) оператор обратного O-дельта-кодирования (способ 3)
=> (декодер) оператор обратного O-дельта-кодирования (способ 3)
[Данные D5] Ур. 22

Всякий раз в этих четырех способах при применении к данным операторов кодирования, как это показано уравнением 21 (ур. 21, соответствующее способу 1), уравнением 22 (ур. 22, соответствующим способу 2), уравнением 23 (ур. 23, соответствующим способу 3) и уравнением 24 (ур. 24, соответствующим способу 4), опционально, возможны попытки применения всех этих способов, поскольку один из них может дать большее снижение энтропии обработанных данных, чем остальные. После оптимизации применения способов в кодере 10 и/или декодере 20, предпочтительно, эти или другие способы применяют до тех пор, пока выбранный способ или способы снижают энтропию по сравнению с требуемым объемом информации в данных. Так, способы 1-4 могут применяться для кодирования числовых значений, при этом определение «числовых значений» включает 1-битные данные, а также недвоичные числа, присутствующие в побитно кодируемых битовых подтоках или в многобитных значениях.

Операция вычисления разности дает разность последовательных числовых значений; и, соответственно, операция суммирования дает сумму последовательных числовых значений. Эти операции, выполняемые в кодере 10, имеют соответствующие им обратные операции в декодере 20. Разности или суммы могут вычисляться на основе текущего входного значения и предыдущего входного или результирующего значения, используемого в качестве значения предсказания. Могут также применяться и другие значения предсказания, при этом для их формирования могут применяться более ранние входные и выходные значения в кодере, при условии, что в декодере при этом может быть выполнена обратная операция.

Ни один из этих способов не обеспечивает значительного сжатия данных в кодере 10 и декодере 20, однако все способы могут эффективно применяться для снижения энтропии, в результате чего другие способы сжатия оказываются способными впоследствии сжимать данные, имеющие меньшую энтропию, более эффективно. Эти другие способы сжатия, опционально, представляют собой по меньшей мере одно из следующего: кодирование Хаффмана, арифметическое кодирование, интервальное кодирование, RLE-кодирование, SRLE-кодирование, кодирование модификатором

энтропии. Однако для всех способов необходима передача нескольких числовых значений, чтобы с их помощью операция кодирования и соответствующая ей обратная операция всегда могли быть выполнены однозначным образом, например, чтобы обеспечивалось сжатие данных без потерь и последующая декомпрессия данных без потерь. Очевидно, кодер 10 и декодер 20 имеют информацию о типе числовых значений, содержащихся во входных данных D1. Предпочтительно, предполагается, что известен диапазон чисел, то есть диапазон, заданный значениями MIN и MAX. В принципе, способы могут во всех случаях работать на основе имеющегося диапазона данных. Числовые значения, необходимые для операций кодирования, включают наименьшее встречающееся числовое значение (lowValue) и наивысшее встречающееся числовое значение (highValue); где lowValue больше или равно MIN, а highValue меньше или равно MAX.

На основе этих значений могут быть получены также и другие необходимые числа. Предпочтительно, эти значения передают в различной форме, а недостающие значения, предпочтительно, вычисляют. Например, если известны два значения из набора ["lowValue", "highValue", «число»], где «число» равно [highValue - lowValue], то третье значение может быть вычислено на их основе. Исключение некоторых значений из данных D2 и затем вычисление их в декодере 20 позволяет получить большую степень сжатия в данных D2.

Помимо этих значений необходимо число P, которое может применяться в качестве предыдущего значения при вычислении первого значения, то есть «предсказание». В качестве числа P, то есть «предсказания», всегда может быть выбрано значение, находящееся между 0 и значением «числа». При этом в рассмотренных выше операциях кодирования должно быть обеспечено значение wrapValue, чтобы в декодере 20, при декодировании данных D2/D3 или D4, могли быть восстановлены исходные данные, а именно, это значение используют для максимально возможного сужения диапазона значений, формируемых операциями кодирования. Однако, это значение "wrapValue" должно быть большим, чем «число», и предпочтительно, оно будет иметь значение «число» +1. Опционально, в зависимости от характера данных D1, как упоминалось выше, первое значение предсказания может быть выбрано равным «0», например, если предполагается, что данные D1 содержат в основном малые значения, а не большие значения; альтернативно, значение первого «предсказания» может быть выбрано равным «числу», если предполагается, что данные D1 содержат в основном большие значения, а не малые значения. В случае если относительно величины значений не было сделано никаких предположений, то, предпочтительно, в качестве значения «предсказания» используют значение "(wrapValue + 1) div 2 + lowValue".

Далее будут описаны примеры операций, выполняемых в вычислительной аппаратуре при реализации вариантов осуществления настоящего изобретения.

В кодере 10 первую прямую операцию вычисления разности, а именно, способ 1, предпочтительно, реализуют следующим образом; для всех значений данных, в программном цикле, вычисляют выходное значение, а именно, «результат», которое соответствует входному, то есть «исходному», значению:

результат = исходное - предсказание

если результат < lowValue, то результат = результат + wrapValue

Наконец, значение предсказания для следующего входного значения назначают равным текущему входному значению, а именно:

предсказание = исходное

В декодере 20 первую обратную операцию вычисления разности, а именно, способ 1, предпочтительно, реализуют следующим образом: для всех значений данных, в программном цикле, вычисляют выходное значение, а именно, «результат», которое соответствует входному, то есть «исходному», значению:

результат = исходное + предсказание

если результата > highValue, то результат = результат – wrapValue

Наконец, значение предсказания для следующего входного значения назначают равным текущему результату, а именно:

предсказание = результат

В кодере 10 вторую прямую операцию вычисления разности, а именно, способ 2, предпочтительно, реализуют следующим образом; для всех значений данных, в программном цикле, вычисляют выходное значение, а именно, «результат», которое соответствует входному, то есть «исходному», значению:

результат = исходное - предсказание

если результат < lowValue, то результат = результат + wrapValue

Наконец, значение предсказания для следующего входного значения назначают равным текущему результату, а именно:

предсказание = результат

В декодере 20 вторую обратную операцию вычисления разности, а именно, способ 2, предпочтительно, реализуют следующим образом: для всех значений данных, в программном цикле, вычисляют выходное значение, а именно, «результат», которое соответствует входному, то есть «исходному», значению:

результат = исходное + предсказание

если результата > highValue, то результат = результат – wrapValue

Наконец, значение предсказания для следующего входного значения назначают равным текущему входному значению, а именно:

предсказание = исходное

В кодере 10 первую прямую операцию суммирования а именно, способ 3, предпочтительно, реализуют следующим образом; для всех значений данных, в программном цикле, вычисляют входное значение, а именно, «результат», которое соответствует входному, то есть «исходному», значению:

результат = исходное + предсказание

если результата > highValue, то результат = результат – wrapValue

Наконец, значение предсказания для следующего входного значения назначают равным текущему входному значению следующим образом:

предсказание = исходное

В декодере 20 первую обратную операцию суммирования а именно, способ 3,

предпочтительно, реализуют следующим образом; для всех значений данных, в программном цикле, вычисляют входное значение, а именно, «результат», которое соответствует входному, то есть «исходному», значению:

результат = исходное - предсказание

если результат < lowValue, то результат = результат + wrapValue

Наконец, значение предсказания для следующего входного значения назначают равным текущему результату, а именно:

предсказание = результат

В кодере 10 вторую прямую операцию суммирования а именно, способ 4, предпочтительно, реализуют следующим образом; для всех значений данных в программном цикле вычисляют входное значение, а именно, «результат», которое соответствует входному, то есть «исходному», значению:

результат = исходное + предсказание

если результата > highValue, то результат = результат – wrapValue

Наконец, значение предсказания для следующего входного значения назначают равным текущему входному значению следующим образом:

предсказание = результат

В декодере 20 вторую обратную операцию суммирования а именно, способ 4, предпочтительно, реализуют следующим образом; для всех значений данных, в программном цикле, вычисляют входное значение, а именно, «результат», которое соответствует входному, то есть «исходному», значению:

результат = исходное - предсказание

если результат < lowValue, то результат = результат + wrapValue

Наконец, значение предсказания для следующего входного значения назначают равным текущему входному значению, а именно:

предсказание = исходное

Такие операции вычисления разности и суммирования, т.е. все четыре способа, применимы также к 1-битным данным, то есть могут применяться побитно, а именно, при реализации вариантов кодера 10 и декодера 20 с применением О-дельта-кодирования. В случае 1-битных данных, следующие значения являются уже известными кодеру 10 и декодеру 20, а именно, MIN = 0, MAX = 1. При этом, предпочтительно, предполагают, что lowValue = MIN = 0, a highValue = MAX = 1. Также, в этом случае «число», соответственно, равно [highValue - lowValue = 1-0=1], а значение wrapValue, предпочтительно, выбирают равным значению «0», поскольку считается, что исключительно 1-битные данные могут иметь только положительные значения, начинающиеся с lowValue = MIN = 0. В случае 1-битных данных способ 1 и способ 3 дают идентичные друг другу результаты кодирования. Аналогично, способ 2 и способ 4 дают идентичные друг другу результаты кодирования. Осведомленность об этом значительно упрощает информацию, которую необходимо передавать в данных D2, поскольку могут быть сделаны допущения по заданным по умолчанию значениям, а именно, необходимо передавать только информацию о том, сколько раз выполняются операции вычисления разности, т.е. способ 1 и способ 2, и выбранное предсказание

(входное значение (способ 1) или результирующее значение 2), чтобы декодер 20 мог быть способен выполнить соответствующую обратную операцию вычисления разности необходимое количество раз при декодировании данных D2, D3 или D4, с целью формирования декодированных данных D5.

Первый пример, который был создан с помощью способа 1 или способа 3, дающего аналогичные выходные данные, допускает также обработку с использованием способа 2 или способа 4, которые также формируют аналогичные выходные данные. Результат, показанный ниже, может быть получен при помощи применения этих способов к данным ур. 1:

0 1 1 0 0 1 0 0 0 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 0 1 0 1 0 1

На этот раз обработанные данные имеют двадцать четыре единицы («1») и тринадцать нулей («0»), то есть энтропия такая же, как и в первом примере, однако количество единиц и нулей поменялось местами. Это случается не всегда, напротив, при использовании различных способов также часто меняется и энтропия. Например, после первых четырех элементов данных способ 1 и/или способ 3 формирует три единицы («1») и один ноль («0»), тогда как в исходных данных и данных, обработанных способом 2 и/или способом 4, присутствуют две единицы («1») и два нуля («0»). Соответственно, в данном случае способ 1 и/или способ 3 дают меньшую энтропию, чем способ 2 и/или способ 4, а также энтропию, меньшую исходной.

В многобитной реализации, если данные D1 содержат значения в диапазоне от -64 до +63, то MIN=-64, а MAX=63. Если принять lowValue = MIN и highValue = MAX, то «число» = 127, а wrapValue, предпочтительно, выбирают равным 128. Однако если данные D1 изменяются случайным образом, значение «предсказания», предпочтительно назначают равным значению [(wrapValue + 1) div 2 + lowValue = 64+(-64)=0].

Следует отметить, что если первое значение равно, например, -1, то первое значение, кодированное с помощью способа 1 и/или способа 2 прямого O-дельта-кодирования, будет равно -1-0= -1, а соответственно, кодированное с помощью 3 и/или способа 4 прямого O-дельта-кодирования: -1+0=-1. Следующие значение могут затем меняться в зависимости от дальнейших изменений данных, например, если вторым значением будет 5, то способ 1 прямого O-дельта-кодирования даст 5-(-1)=6, способ 2 прямого O-дельта-кодирования даст 5-(-1)=6, способ 3 прямого O-дельта-кодирования даст 5+(-1) =4, а способ 4 прямого O-дельта-кодирования даст 5+(-1)=4. Декодер 20 в этом случае будет способен формировать, в качестве первого значения при использовании способа 1 и/или способа 2 обратного O-дельта-кодирования, -1+0 =-1, и при использовании способа 3 и/или способа 4 обратного O-дельта-кодирования, -1-0 =-1. Соответственно, второе значение, полученное с помощью способа 1 обратного O-дельта-кодирования, будет равно 6+(-1)=5, с помощью способа 2 обратного O-дельта-кодирования, оно будет равно 6+(-1)=5, с использованием способа 3 обратного O-дельта-кодирования, оно будет равно 4-(-1)=5, и с помощью способа 4 обратного O-дельта-кодирования, оно будет равно 4-(-1)=5.

Данное решение может затем быть оптимизировано, если диапазон чисел фактически содержит только значения между -20 и +27. В данном примере возможна передача, например, lowValue = -20, и highValue = 27. Если передают оба значения, то можно вычислить, что «количество» = 47, и wrapValue в этом случае, предпочтительно, выбирают равным 48. Теперь можно вычислить значение «предсказания», которое равно 48 div 2+(-20)=4. Далее, предыдущий пример дал бы, например, для значения -1, при использовании способа 1 или способа 2 O-дельта-кодирования кодирования: -1-4

=-5, а при использовании способа 3 или способа 4 О-дельта-кодирования кодирования: -1+4=3. Аналогично, второе значение для способов О-дельта-кодирования кодирования было бы равным (5-(-1))=6, (5-(-5))=10, (5+(-1))=4 (5+3)=8. Декодирование снова выполняется корректно и дает первое значение для способа 1 и/или способа 2, равное -5+4=-1, а для способа 3 и/или способа 4, равное 3-4 = -1. Соответственно, вторые значения для различных способов будут декодированы как (6+(-1))=5, (10+(-5))=5, (4-(-1))=5, и (8-3)=5.

Следует отметить, что все значения в рассмотренных выше примерах находятся внутри диапазона, а именно, от -64 до 63 или от -20 до +27, поэтому для данных примеров значений не требуется ввод поправки, однако если какая-либо из положительных или отрицательных разностей является достаточно большой, то необходимо вносить в данные поправку, при помощи приведенных уравнений 21-24 (ур. 21-24), чтобы результирующие значения находились внутри диапазона. Следует отметить, что под поправкой в настоящем документе понимается значение циклического перехода.

Если известно lowValue, кодированные значения, предпочтительно, конфигурируют так, чтобы они начинались с 0 и заканчивались значением «числа», что позволяет упростить таблицу кодирования, которую необходимо передавать из кодера 10 в декодер 20 вместе с энтропийно-кодированными данными D3. Эту операцию называют пост-смещением, при этом значение пост-смещения должно быть удалено из значений кодированных данных после энтропийного декодирования до выполнения операции обратного О-дельта-кодирования над данными D4.

Как отмечалось выше, возможна также реализация смещения с использованием функциональности предварительного смещения, при которой исходные входные данные (D1) преобразуют в положительные элементы, содержащие значения от нуля до «числа», уже непосредственно перед выполнением способа О-дельта-кодирования. Также, в такой ситуации, передача информации, необходимой для этой операции, предпочтительно, должна выполняться таким образом, чтобы способы «предварительного смещения» и способ О-дельта-кодирования не передавали повторно одну и ту же информацию, или опускали информацию, уже известную в другом способе. Для получения корректных выходных данных D5 результат предварительного смещения должен быть устранен из декодированных данных перед выполнением операции обратного О-дельта-кодирования.

В пределах сущности и объема, определенных приложенной формулой изобретения, возможны модификации вариантов осуществления настоящего изобретения, приведенных в предшествующем описании. Такие выражения как «включающий», «содержащий», «охватывающий», «состоящий из», «имеющий», «представляющий собой», которые использованы в описании и в формуле настоящего изобретения, должны пониматься как не исключающие, то есть допускающие также присутствие объектов, компонентов или элементов, явно не упомянутых. Использование единственного числа может также пониматься как относящееся к множественному числу. Числовые обозначения на приложенных чертежах, приведенные в скобках, имеют целью упрощение понимания пунктов формулы изобретения и не должны пониматься как ограничивающие изобретение, определенное этими пунктами формулы изобретения.

Формула изобретения

1. Кодер (10) для кодирования входных данных (D1), включающих последовательность числовых значений, с получением соответствующих кодированных выходных данных (D2 или D3), включающий схему обработки данных для применения

к входным данным (D1) одной из форм разностного и/или суммирующего кодирования для формирования одной или более соответствующих кодированных последовательностей, при этом одна или более соответствующих кодированных последовательностей подвергается циклическому переходу относительно максимального значения и/или циклическому переходу относительно минимального значения для формирования кодированных выходных данных (D2 или D3).

2. Кодер (10) по п. 1, отличающийся тем, что схема обработки данных выполнена с возможностью анализа входных данных (D1) и/или одной или более соответствующих кодированных последовательностей для вычисления одного или более значений смещения, минимальных или максимальных значений для применения к одной или более соответствующим кодированным последовательностям для использования при формировании кодированных выходных данных (D2 или D3).

3. Кодер (10) по п. 2, отличающийся тем, что одно или более значений смещения имеют значение «0».

4. Кодер (10) по п. 1, выполненный с возможностью обработки числовых значений, включающих одно или более 1-битных значений, и с возможностью кодирования входных данных (D1) побитно.

5. Кодер по п. 1, отличающийся тем, что одна или более соответствующих кодированных последовательностей представляют изменения в последовательных значениях входных данных (D1).

6. Кодер (10) по п. 1, выполненный с возможностью разбиения входных данных (D1) на множество фрагментов данных, которые кодируют отдельно.

7. Кодер (10) по п. 6, выполненный с возможностью избирательного применения кодирования к фрагментам данных, только когда при помощи этого может быть достигнуто сжатие данных в кодированных выходных данных (D2 или D3).

8. Кодер (10) по п. 1, отличающийся тем, что кодер (10) выполнен с возможностью применения заданного по умолчанию первого значения предсказания для последовательности значений предсказания, которые применяют для создания выходных кодированных данных (D2 или D3).

9. Кодер (10) по п. 8, отличающийся тем, что заданным по умолчанию первым значением предсказания является по меньшей мере одно из «0», maxValue/2, maxValue/2 + lowValue, highValue - lowValue.

10. Кодер (10) по п. 8, отличающийся тем, что он выполнен с возможностью применения дополнительного кодирования для формирования кодированных выходных данных (D2), при этом дополнительное кодирование включает по меньшей мере одно из следующего: кодирование длин серий (RLE), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование.

11. Кодер (10) по п. 6, выполненный с возможностью разбиения входных данных (D1) на множество фрагментов в зависимости от длины серий идентичных друг другу битов в них, что является эффективным для кодирования с использованием кодирования длин серий (RLE), кодирования Хаффмана, кодирования переменной длины (VLE), интервального кодирования и/или арифметического кодирования.

12. Кодер (10) по любому из предшествующих пунктов, отличающийся тем, что схема обработки данных реализована с использованием вычислительной аппаратуры, выполненной с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

13. Способ использования кодера (10) для кодирования входных данных (D1), включающих последовательность числовых значений, с получением соответствующих

кодированных выходных данных (D2 или D3), включающий:

(a) применение к входным данным (D1) одной из форм разностного и/или суммирующего кодирования для получения одной или более соответствующих кодированных последовательностей; при этом

(b) одну или более соответствующих кодированных последовательностей подвергают операции циклического перехода относительно максимального значения и/или циклического перехода относительно минимального значения для формирования кодированных выходных данных (D2 или D3).

14. Способ по п. 13, включающий использование схемы обработки данных для анализа входных данных (D1) и/или одной или более соответствующих кодированных последовательностей для вычисления одного или более значений смещения, минимальных или максимальных значений для применения к одной или более соответствующим кодированным последовательностям для использования при формировании кодированных выходных данных (D2 или D3).

15. Способ по п. 14, включающий использование значения «0» для одного или более значений смещения.

16. Способ по п. 14, включающий обработку числовых значений, включающих одно или более 1-битных значений, при этом кодер (10) выполнен с возможностью побитного кодирования входных данных (D1).

17. Способ по п. 13, отличающийся тем, что одна или более соответствующих кодированных последовательностей представляют изменения в последовательных значениях входных данных (D1).

18. Способ по п. 13, включающий использование схемы обработки данных для разбиения входных данных (D1) на множество фрагментов данных, которые после этого кодируют отдельно.

19. Способ по п. 18, включающий избирательное применение упомянутой формы кодирования к фрагментам данных, только когда при помощи этого может быть достигнуто сжатие данных в кодированных выходных данных (D2 или D3).

20. Способ по п. 13, отличающийся тем, что способ включает применение заданного по умолчанию первого значения предсказания для последовательности значений предсказания, которые применяют для создания кодированных выходных данных (D2 или D3).

21. Способ по п. 20, отличающийся тем, что заданным по умолчанию первым значением предсказания является по меньшей мере одно из «0», maxValue/2, maxValue/2 + lowValue, highValue - lowValue.

22. Способ по любому из пп. 13-21, включающий применение в нем дополнительного кодирования для формирования кодированных выходных данных (D2 или D3), при этом дополнительное кодирование включает по меньшей мере одно из следующего: кодирование длин серий (RLE), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование.

23. Способ по любому из пп. 13-21, включающий реализацию схемы обработки данных с использованием вычислительной аппаратуры, выполненной с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

24. Декодер (20) для декодирования кодированных данных (D2 или D3) с получением соответствующих декодированных выходных данных (D5), включающий схему обработки данных для обработки одной или более частей кодированных данных (D2 или D3), при этом схема обработки данных выполнена с возможностью применения

одного из видов разностного и/или суммирующего декодирования к одной или более соответствующих кодированных последовательностей упомянутых одной или более частей, при этом одна или более кодированные последовательности подвергаются операции циклического перехода относительно максимального значения и/или циклического перехода относительно минимального значения, для формирования декодированных выходных данных (D5).

25. Декодер (20) по п. 24, выполненный с возможностью декодирования кодированных данных (D2 или D3), включающих одно или более 1-битных значений, при этом декодер (20) выполнен с возможностью побитного декодирования кодированных данных (D2 или D3).

26. Декодер (20) по п. 24, отличающийся тем, что схема обработки данных выполнена с возможностью вычисления одного или более значений смещения, минимального или максимального значений для применения к одной или более кодированным последовательностям для использования при формировании декодированных выходных данных (D5).

27. Декодер (20) по п. 26, отличающийся тем, что одно или более значений смещения имеют значение «0».

28. Декодер (20) по п. 24, отличающийся тем, что одна или более соответствующих кодированных последовательностей представляют изменения, закодированные в кодированных данных (D2 или D3).

29. Декодер (20) по п. 24, отличающийся тем, что схема обработки данных выполнена с возможностью применения к данным, обрабатываемым при ее помощи, преобразования, обратного по меньшей мере одному из следующего: кодирование длин серий (RLE), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование.

30. Декодер (20) по п. 24, отличающийся тем, что схема обработки данных выполнена с возможностью предполагать задаваемое по умолчанию первое значение предсказания в последовательности данных, декодируемых этой схемой.

31. Декодер (20) по п. 30, отличающийся тем, что упомянутое заданное по умолчанию значение имеет значение «0».

32. Декодер (20) по любому из пп. 24-31, отличающийся тем, что схема обработки данных реализована с использованием вычислительной аппаратуры, выполненной с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

33. Способ использования декодера (20) для декодирования кодированных данных (D2 или D3) с получением соответствующих декодированных выходных данных (D5), включающий:

а) использование схемы обработки данных для обработки одной или более частей кодированных данных (D2 или D3), при этом

б) схема обработки данных выполнена с возможностью применения одной из форм разностного и/или суммирующего декодирования к одной или более соответствующим кодированным последовательностям упомянутых одной или более частей, при этом одну или более кодированных последовательностей подвергают операции циклического перехода относительно максимального значения и/или циклического перехода относительно минимального значения, для формирования декодированных выходных данных (D5).

34. Способ по п. 33, включающий использование декодера (20) для декодирования кодированных данных (D2 или D3), включающих одно или более 1-битных значений,

при этом декодер (20) выполнен с возможностью побитного декодирования кодированных данных (D2 или D3).

35. Способ по п. 33, включающий использование схемы обработки данных для вычисления одного или более значения смещения, минимального или максимального значения для применения к одной или более соответствующим кодированным последовательностям для использования при формировании декодированных выходных данных (D5).

36. Способ по п. 35, отличающийся тем, что одно или более значений смещения имеют значение «0».

37. Способ по п. 33, отличающийся тем, что одна или более соответствующих кодированных последовательностей представляют изменения в последовательных значениях, закодированных в кодированные данные (D2 или D3).

38. Способ по п. 33, отличающийся тем, что способ включает использование схемы обработки данных для применения к данным, обрабатываемым с ее помощью, преобразования, обратного по меньшей мере одному из следующего: кодирование длин серий (RLE), кодирование переменной длины (VLC), кодирование Хаффмана, арифметическое кодирование, интервальное кодирование.

39. Способ по п. 33, отличающийся тем, что способ включает использование схемы обработки данных для предположения о задаваемом по умолчанию первом значении предсказания в последовательности данных, декодируемых этой схемой.

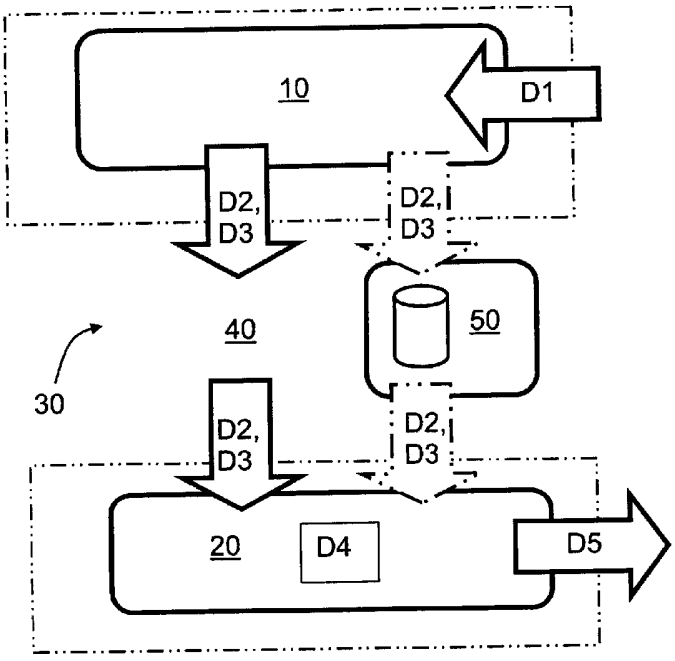
40. Способ по п. 39, отличающийся тем, что упомянутое заданное по умолчанию значение имеет значение «0».

41. Способ по любому из пп. 33-40, отличающийся тем, что способ включает реализацию схемы обработки данных с использованием вычислительной аппаратуры, выполненной с возможностью исполнения одного или более программных продуктов, записанных на машиночитаемом носителе для хранения данных.

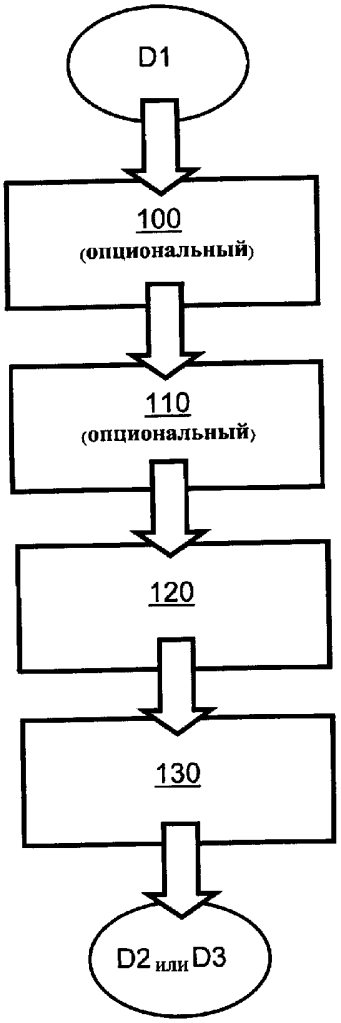
42. Кодек (30), включающий по меньшей мере один кодер (10) по п. 1 для кодирования входных данных (D1) с получением соответствующих кодированных данных (D2) и по меньшей мере один декодер (20) по п. 24 для декодирования кодированных данных (D2 или D3) с получением декодированных данных (D5).

43. Машиночитаемый носитель для хранения данных, на котором записан программный продукт, исполняемый на вычислительном оборудовании для выполнения способа кодирования данных по любому из пп. 13-23.

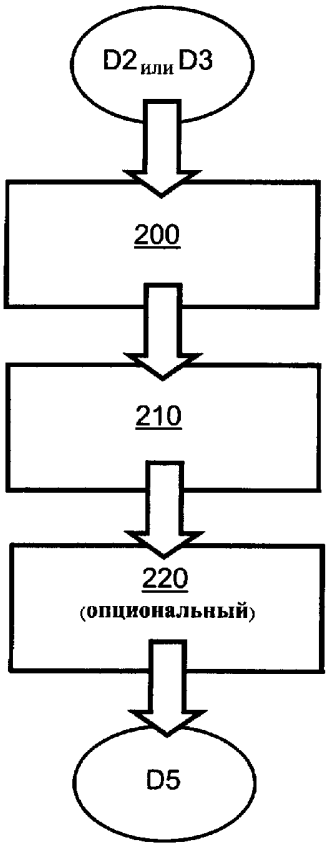
44. Машиночитаемый носитель для хранения данных, на котором записан программный продукт, исполняемый на вычислительном оборудовании для выполнения способа декодирования данных по любому из пп. 33-41.



Фиг.1



Фиг.2



Фиг.3