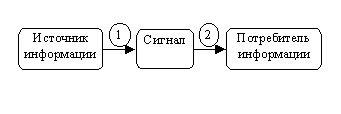
1. Теорія іфнормації та її застосування в різних галузях науки і техніки

Информация – это сведения об окружающем мире (объекте, процессе, явлении, событии), которые являются объектом преобразования (включая хранение, передачу и т.д.) и используются для выработки поведения, для принятия решения, для управления или для обучения.

Как следует из этого определения, с информацией всегда связывают три понятия:

* источник информации – тот элемент окружающего мира (объект, процесс, явление, событие), сведения о котором являются объектом преобразования. Так, источником информации, которую в данный момент получает слушатель данной лекции, является информатика как сфера человеческой деятельности;
* потребитель информации – тот элемент окружающего мира, который использует информацию (для выработки поведения, для принятия решения, для управления или для обучения).
* сигнал – материальный носитель, который фиксирует информацию для переноса ее от источника к потребителю. В данном случае сигнал носит электронный характер. Если же студент возьмет данное пособие в библиотеке, то та же информация будет иметь бумажный носитель. Будучи прочитанной и запомненной студентом, информация приобретет еще один носитель – биологический, когда она ―записывается‖ в память обучаемого.

Взаимосвязь введенных понятий показана на рисунке:



Сигнал является материальным носителем информации, которая передается от источника к потребителю. Он может быть дискретным и непрерывным (аналоговым).

Дискретный сигнал слагается из счетного множества (т.е. такого множества, элементы которого можно пересчитать) элементов (говорят – информационных элементов).

Например, дискретным является сигнал ―кирпич‖. Он состоит из следующих двух элементов (это синтаксическая характеристика данного сигнала): красного круга и белого прямоугольника внутри круга, расположенного горизонтально по центру.

Непрерывный сигнал – отражается некоторой физической величиной, изменяющейся в заданном интервале времени, например, тембром или силой звука. В виде непрерывного сигнала представлена настоящая информация для тех студентов – потребителей, которые посещают лекции по информатике и через звуковые волны (иначе говоря, голос лектора), носящие непрерывный характер, воспринимают материал.

1. Кількісні оцінки інформації

В качестве основной характеристики сообщения теория информации принимает величину, называемую *количеством* *информации*. *Это понятие не затрагивает смысла и важности передаваемого сообщения, а связано со степенью его неопределенности.*Количество информации – мера неопределённости, «снятой»/устраненной при получении сообщения.Количество информации в сообщении о некотором событии существенно зависит от вероятности этого события.Пусть имеется m качественных признаков сообщения (количество символов алфавита, количество уровней квантования, m называется мощностью алфавита). Пусть n - число элементов сообщения. Тогда существует mn различных сообщений длиной n.

Меры количества информации

*1-я количественная мера информации (элементарная)*

Количество информации I определяется количеством сообщений, которые можно составить из m символов алфавита при длине сообщения n, I=mn Недостаток этой меры – неаддитивность (непропорциональность количества информации и длины

сообщения). Логично предположить, что сообщение, имеющее, например, в два раза большую длину, несет в два раза большее количество информации.

*2-я количественная мера – мера Р.Хартли*

Ральф Винтон Лайон Хартли (англ. Ralph Vinton Lyon Hartley, 30 ноября 1888 – 1 мая 1970) – американский исследователь в области электроники; изобрёл осциллятор Хартли и преобразователь Хартли; внес вклад в основания теории информации, введя в 1928 логарифмическую меру информации *H* = *K*ln*M*, которая называется *хартлиевским количеством* *информации*.

I = n × logm

Отсюда возникла и двоичная единица информации. Сообщение должно иметь минимум один символ: nmin=1. Алфавит должен иметь минимум два элемента: mmin=2. Вот в таком сообщении и содержится минимально возможное количество информации.

Imin = nmin∙log mmin

При использовании двоичного логарифма Imin = 1 (бит, от англ. binary digit). Поэтому в теории информации используется логарифм по основанию 2

I = n∙log2m.

Пусть алфавит источника сообщений состоит из m знаков, каждый из которых может служить элементом сообщения. Количество N возможных сообщений длины n равно числу перестановок с неограниченными повторениями: N = mn Если для получателя все N сообщений от источника являются равновероятными, то получение конкретного сообщения равносильно для него случайному выбору одного из N сообщений с вероятностью 1/N. *Ясно, что чем больше* N*, тем большая степень неопределенности характеризует этот выбор и тем более информативным можно считать сообщение*. Поэтому число N могло бы служить мерой информации. Однако, с позиции теории информации, естественно наделить эту меру свойствами *аддитивности*,т.е.определить ее так,чтобы она была пропорциональнадлине сообщения (например, при передаче и оплате сообщения - телеграммы, важно не ее содержание, а общее число знаков). *В качестве меры неопределенности выбора состояния источника с равновероятными состояниями принимают логарифм числа состояний:* I = log N = log mn = n log m. *Эта логарифмическая функция характеризует количество информации.* В принципе безразлично, какое основание логарифма использовать для определения количества информации и энтропии, т. к. в силу соотношения loga m =loga b logb m переход от одного основания логарифма к другому сводится лишь к изменению единицы измерения. Так как современная информационная техника базируется на элементах, имеющих два устойчивых состояния, то обычно выбирают основание логарифма равным двум, т.е. энтропию выражают как: H0 = log2 m.

Тогда *единицу количества информации* на один элемент сообщения называют *двоичной единицей* или *битом*. При этом единица неопределенности (двоичная единица или бит) представляет собой неопределенность выбора из двух равновероятных событий *(bit* — сокращение от англ. *binary* *digit* —двоичная единица) Так как из log2 m = 1 следует m = 2, то ясно, что *1* *бит* *-* *это количество* *информации, которым характеризуется один двоичный элемент при равновероятных состояниях 0 и 1.* Двоичное сообщение длины n содержит n бит информации. Единица количества информации, равная 8 битам, называется байтом. Если основание логарифма выбрать равным десяти, то энтропия выражается в десятичных единицах на элемент сообщения - дитах, причем 1дит = log102 бит = 3,32 бит.

*3-я количественная мера – мера К.Шеннона*

Клод Элвуд Шеннон (1916-2001) – американский инженер, член Национальной АН США, профессор Массачусетского технологического института, в 1948 г. издал книгу «Математическая теория связи» с изложением основ теории информации. Будучи студентом Массачусетского технологического института, который он окончил в 1936 году, Шеннон специализировался одновременно и в математике, и в электротехнике. В 1940 году он защитил диссертацию, в которой доказал, что работу обычных переключателей и реле в электрических схемах можно представить посредством булевой алгебры. Сейчас булева алгебра лежит в основе современной цифровой схемотехники, но тогда применение к технике методов английского ученого Джорджа Буля было делом революционным. В 1941 году 25-летний Клод Шеннон поступил на работу в Bell Laboratories, где, помимо всего прочего, прославился тем, что катался на одноколесном велосипеде по коридорам лаборатории, одновременно жонглируя четырьмя мячиками. В годы войны он занимался разработкой криптографических систем, и позже это помогло ему открыть методы кодирования с коррекцией ошибок. Группа инженеров под руководством К.Шеннона сумела декодировать систему шифрования Энигма, что помогло союзникам быть в курсе гитлеровских планов. Кстати, в те же сороковые годы Шеннон занимался конструированием летающего диска на ракетном двигателе. В работах 1957-61 годов Шеннон построил свою теорию пропускной способности каналов связи.Количественная мера Шеннона получена из следующих соображений. Если символ появляется в сообщении с вероятностью 1 (на определенном месте), то такой символ никако

информации не несет. В случае если любой из m символов алфавита равновероятен p=1/m, то количество информации, содержащееся в сообщении из одного элемента определяется так:

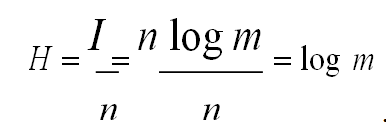
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | p = | 1 | |  |  | 1 |  |
| I0 = 1log m = | m | |  | = log | = -log p |
|  |  |
|  | 1 | |  | p |
|  |  |  |  |  |
|  | m = | |  |  |  |  |  |
| p |  |  |  |  |
|  |  |  |  |  |  |  |  |

Если p – вероятность символа, то I0 – информация в сообщении из одного символа. Но неопределенно, что такое «один символ». Сколько, например, символов в сообщениях «МАМА МЫЛА РАМУ» или «Над Киевом безоблачное небо»? Очевидно, что любое сообщение можно представить в виде одного символа соответствующего алфавита. Количество информации не должно зависеть от способа выбора алфавита. Соответственно, эту формулу можно распространить на произвольные сообщения, учитывая, что под P будем понимать вероятность сообщения.

I = - log2P

В сообщении из n элементов вероятность каждого символа равна p. Вероятность n символов равна Р = pn . С учетом того, что для равновероятных символов p=1/m имеем

Количество информации, приходящееся на один элемент сообщения (знак, букву), называется энтропией:



1. Ентропія дискретних повідомлень

Энтропия – мера неопределенности случайного состояния некоторой системы. Мы рассматриваем информационные системы, то есть системы, воспринимающие, хранящие, перерабатывающие и использующие информацию. Нормальное функционирование подобных систем – это прием-передача информационных сообщений.Для целей теории информации мы определим энтропию как среднее количество информации, приходящееся на одно сообщение в ансамбле сообщений (или на один символ в отдельном сообщении). Иначе говоря, энтропия – это математическое ожидание количества информации в сообщении.Пусть информационная система может порождать ансамбль (алфавит) сообщений a1, a2,…,am. Вероятности каждого сообщения: P(a1), P(a2), …,P(am). Так как вероятности сообщений не одинаковы, то они несут разное количество информации.

I(ai) = - log2 P(ai).

Среднее количество информации (математическое ожидание):

Совершенно аналогично вводится энтропия сообщений:



Энтропия не зависит от конкретного сообщения. Это характеристика информационной системы (источника сообщений или канала передачи сообщений). Энтропия в таком виде является априорной характеристикой и может быть вычислена до эксперимента, если известна статистика сообщений. Энтропия характеризует неопределенность ситуации до передачи сообщения, поскольку заранее не известно, какое сообщение из ансамбля будет передано. Чем больше энтропия, тем сильнее неопределенность и тем большую информацию в среднем несет одно сообщение источника. Сравнивая формулы (8) и (6) видим, что

I = n∙H

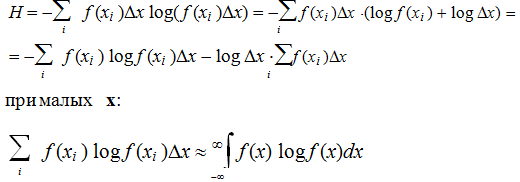
*Свойства энтропии*

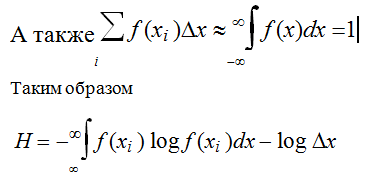
1. Энтропия принимает значение, равное 0, только в случае детерминированного источника сообщений системы.
2. Энтропия - величина неотрицательная и ограниченная.
3. Энтропия системы, имеющей m равновероятных состояний, максимальна и равна log2m*.*
4. Совместная энтропия независимых источников сообщений равна сумме энтропий.
5. Ентропія неперервних повідомлень

Рассмотрим систему, где качественные признаки состояния изменяются непрерывно (непрерывный сигнал). Вероятность нахождения системы в состоянии х (т.е. сигнал принимает значение х) характеризуется плотностью вероятности f(x). Чтобы найти энтропию такого сообщения, разбиваем диапазон возможного изменения сигнала на дискреты размером ∆x. Вероятность нахождения системы в i-й дискрете равна

P(xi) = f(xi)∙Dx

Тогда энтропия системы вычисляется так:

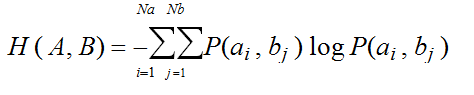




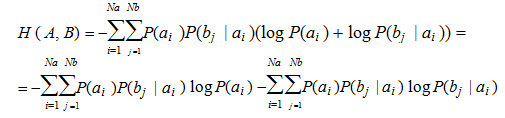
При уменьшении ∆х Н стремится к ∞. Это естественно, т.к. чем точнее мы хотим задать состояние системы, тем большую степень неопределенности мы должны устранить. Дифференциальная энтропия не является мерой количества информации, хотя и характеризует степень неопределенности, присущую источнику.

1. умовна ентропія

Пусть источник **А** порождает ансамбль **Na** сообщений **(a1, a2,…,** **aNa),** источник **B** порождает ансамбль **Nb** сообщений **(b1, b2,…,** **bNb)** и источники **зависимы**.Общий алфавит источников представляет собой множество пар вида **(ai, bj),** общая мощность алфавита: **NaNb**.

Энтропия сложной информационной системы (из двух источников) равна

Поскольку **A** и **B** зависимы, то **P(ai,bj) = P(ai)∙P(bj|ai)**, a **log P(ai,bj) = log P(ai) + log P(bj|ai)**. Подставив это в выражение для энтропии сложной системы, получаем:

В первом слагаемом индекс **j** имеется только у **B,** изменив порядок

**Nb**

суммирования, получим член вида: **P(bj****| ai****)**,который равен**1**поскольку

**j****1**

характеризует достоверное событие (какое-либо сообщений **bj** в любом случае реализуется). Следовательно, первое слагаемое оказывается равным:

*Na*

*P*(*ai* ) log *P*(*ai* ) *H* ( *A*)

*i*1

*Nb*

Во втором слагаемом члены вида *P*(*bj*|*ai*) log*P*(*bj*|*ai*)*H*(*B*|*ai*)имеют

*j* 1

смысл энтропии источника **B** при условии, что реализовалось сообщение **ai** – будем называть ее **частной условной энтропией**. Если ввести данное понятие и использовать его обозначение, то второе слагаемое будет иметь вид:

|  |  |
| --- | --- |
| *Na* |  |
| *P*(*ai* )*H* (*B* | *ai* ) *H* (*B* | *A*), |

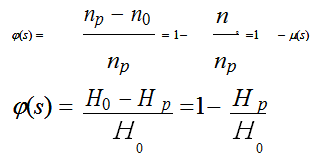
*i*1

|  |
| --- |
| **H(A, B) = H(A) + H(B|A)** |

где **H(B|A)** есть **общая условная энтропия** источника **В** относительно источника **А**. Окончательно получаем для энтропии сложной системы:

1. надмірність повідомлення та методи її зменшення

Коэффициент избыточности выражается так

 Очевидно, что троичный алфавит является более экономичным, чем двоичный.

1. Основні принципи оптимального кодування

Кодирование, которое осуществляет удаление или уменьшение избыточности из закодированных сообщений,

называется *эффективным*.

Возможность эффективного кодирования основана на теореме Шеннона о кодировании, согласно которой:

*Минимальное среднее количество элементов на выходе кодирующего устройства, соответствующее одному символу дискретного сообщения, можно сделать сколь угодно близким к максимальной энтропии источника за счет выбора соответствующего способа кодирования .*

Эффективное кодирование осуществляется с применением неравномерных кодов, в которых более короткие кодовые комбинации соответствуют более вероятным символам сообщения, а более длинные — менее вероятным символам.

*Оптимальные коды -* коды с практически нулевойизбыточностью. Оптимальные коды имеют минимальную среднюю длину кодовых слов - L. Верхняя и нижняя границы *L* определяются из неравенства



где *Н* *-* энтропия первичного алфавита, *т* *-* число качественных признаков вторичного алфавита.

При построении оптимальных кодов наибольшее распространение нашли методики Шеннона—Фано и Хаффмена.

1. **методика шенона фено**

Алгоритм Шеннона — Фано — один из первых алгоритмов сжатия, который впервые сформулировали американские учёные Шеннон и Роберт Фано.

Этот метод требует упорядочения исходного множества символов по не возрастанию их частот. Затем выполняются следующие шаги:

а) список символов делится на две части (назовем их пе вой и вто ой частями) так, что ы суммы частот о еих частей (назовем их Σ1 и Σ2) ыли точно или п име но авны. В случае, когда точного авенства достичь не удается, азница между суммами должна ыть минимальна;

б) кодовым ком инациям пе вой части дописывается 1, кодовым ком инациям вто ой части дописывается 0;

в) анализи уют пе вую часть: если она соде жит только один символ, а ота с ней заканчивается, – считается, что код для ее символов пост оен, и выполняется пе еход к шагу г) для пост оения кода вто ой части. Если символов ольше одного, пе еходят к шагу а) и п оцеду а повто яется с пе вой частью как с самостоятельным упо ядоченным списком;

г) анализи уют вто ую часть: если она соде жит только один символ, а ота с ней заканчивается и выполняется о ащение к оставшемуся списку (шаг д). Если символов ольше одного, пе еходят к шагу а) и п оцеду а повто яется со вто ой частью как с самостоятельным списком;

д) анализи уется оставшийся список: если он пуст – код пост оен, а ота заканчивается. Если нет, – выполняется шаг а).

Нетрудно убедиться, что коды, получаемые в результате применения методики Шеннона — Фано или Хаффмена, являются префиксными

1. Метод хафменна

|  |  |
| --- | --- |
| Этот метод имеет два преимущества по сравнению с методом Шеннона-Фано: он устраняет неоднозначность кодирования, возникающую из-за примерного авенства сумм частот при разделении списка на две части (линия деления проводится неоднозначно), и имеет, вообщем случае, | большую |
| эффективность кода. |  |
| Исходное множество символов упорядочивается | по не |
| возрастанию частоты и выполняются следующие шаги: |  |
| 1) объединение частот: |  |

* две последние частоты списка складываются, а соответствующие символы исключаются из списка;
* оставшийся после исключения символов список пополняется суммой частот и вновь упорядочивается;
* предыдущие шаги повторяются до тех пор, пока не получится единица в результате суммирования и список не уменьшится до одного символа;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2) построение кодового дерева: | | |  |  |
| · | строится двоичное кодовое дерево, корнем его является | | | |
| вершина, полученная в результате объединения частот, | | | | равная 1; |
| листьями | – | исходные вершины; | остальные | вершины |
| соответствуют | | либо суммарным, либо | исходным | частотам, |

причем для каждой вершины левая подчиненная вершина соответствует большему слагаемому, а правая – меньшему; е а дерева связывают вершины-суммы с вершинами-слагаемыми. Структра дерева показывает, как происходило объединение частот;

* єлементы дерева кодируются: каждое левое кодиуется единицей, каждое правое – нулем;
* формирование кода: для получения кодов листьев (исходных кодируемых символов) продвигаются от корня к нужной вершине.

1. Оцінка ефективності оптимальних нерівномірних кодів

Построенные по указанным выше (либо подобным) методикам коды с неравномерным распределением символов, имеющие минимальную среднюю длину кодового слова,называю *оптимальным,* *неравномерным,* *кодами* (ОНК). Равномерные коды могут быть оптимальными только для передачи сообщений с равновероятным распределением символов первичного алфавита, при этом число символов первичного алфавита должно быть равно целой степени числа, равного количеству качественных признаков вторичного алфавита, а в случае двоичных кодов - целой степени двух. Максимально эффективными будут те ОНК, у которых



*(т -* число качественных признаков вторичного алфавита).Для двоичных кодов



так как log22 = 1. Очевидно, что это равенство удовлетворяется при условии, что длина кода во вторичном алфавите

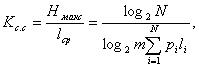
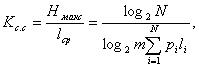


Величина *li* точно равна *Н,* если , где *п* *-* любое целое число. Если *п* не является целым числом для всех значений

букв первичного алфавита, то  и, согласно основной теореме кодирования, средняя длина кодового слова приближается к энтропии источника сообщений по мере укрупнения кодируемых блоков.

Эффективность ОНК. оценивают при помощи

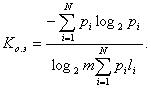
*коэффициента статистического сжатия*



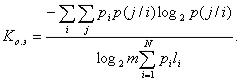
который характеризует уменьшение количества двоичных знаков на символ сообщения при применении ОНК по сравнению с применением методов нестатистического кодирования и

*коэффициента относительной эффективности*



который показывает, насколько используется статистическая избыточность передаваемого сообщения.Для наиболее общего случая неравновероятных и взаимонезависимых символов

Для случая неравновероятных и взаимозависимых символов

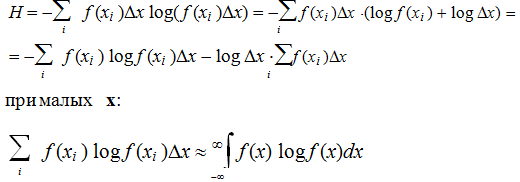


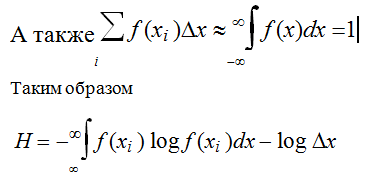
1. Оцінка ентропії неперервних повідомлень

Рассмотрим систему, где качественные признаки состояния изменяются непрерывно (непрерывный сигнал). Вероятность нахождения системы в состоянии х (т.е. сигнал принимает значение х) характеризуется плотностью вероятности f(x). Чтобы найти энтропию такого сообщения, разбиваем диапазон возможного изменения сигнала на дискреты размером ∆x. Вероятность нахождения системы в i-й дискрете равна

P(xi) = f(xi)∙Dx

Тогда энтропия системы вычисляется так:





При уменьшении ∆х Н стремится к ∞. Это естественно, т.к. чем точнее мы хотим задать состояние системы, тем большую степень неопределенности мы должны устранить. Дифференциальная энтропия не является мерой количества информации, хотя и характеризует степень неопределенности, присущую источнику.

1. Оцінка ентропії складних повідомлень
2. Основні методи стиснення данних та їх характеристика

Сти́снення да́них ([англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) *data compression*) — це процедура перекодування [даних](https://uk.wikipedia.org/wiki/%D0%94%D0%B0%D0%BD%D1%96), яка проводиться з метою зменшення їхнього обсягу, розміру, об'єму. Стиснення базується на усуненні надлишку інформації, яка міститься у вихідних даних. Наприклад, повторення в [тексті](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82) фрагментів (наприклад, слів [природної](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D1%80%D0%BE%D0%B4%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) або машинної мови). Подібний надлишок зазвичай усувається заміною повторюваних послідовностей коротшим значенням (кодом). Інший вид надлишковості пов'язаний з тим, що деякі значення в даних, що стискаються, трапляються частіше інших, при цьому можна замінювати дані, що часто трапляються, коротшими кодами, а ті, що рідко, довшими (ймовірнісне стиснення). Стиснення даних, які не мають властивості надлишку (наприклад випадковий [сигнал](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%B3%D0%BD%D0%B0%D0%BB) чи [шум](https://uk.wikipedia.org/wiki/%D0%A8%D1%83%D0%BC)), неможливе. Також, зазвичай, неможливо стиснути зашифровану інформацію.

Види стиснення:

* [Стиснення без втрат](https://uk.wikipedia.org/wiki/%D0%A1%D1%82%D0%B8%D1%81%D0%BD%D0%B5%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B5%D0%B7_%D0%B2%D1%82%D1%80%D0%B0%D1%82) — можливо відновлення вихідних даних без спотворень.
* [Стиснення зі втратами](https://uk.wikipedia.org/wiki/%D0%A1%D1%82%D0%B8%D1%81%D0%BD%D0%B5%D0%BD%D0%BD%D1%8F_%D0%B7_%D0%B2%D1%82%D1%80%D0%B0%D1%82%D0%B0%D0%BC%D0%B8) — відновлення можливе з незначними спотвореннями.

Стиснення без втрат використовується при обробці та збереженні комп'ютерних програм і даних.

Для деяких типів даних спотворення не припустимі в принципі. У їх числі:

* символічні дані, зміна яких неминуче призводить до зміни їх семантики: програми та їх вихідні тексти, виконавчі масиви тощо;
* життєво важливі дані, зміни в яких можуть призвести до критичних помилок: наприклад, одержувані з медичної вимірювальної апаратури або контрольних приладів літальних, космічних апаратів тощо;
* багаторазово піддані стисненню і відновленню проміжні дані при багатоетапній обробці графічних, звукових і відеоданих.
* Стиснення зі втратами зазвичай застосовується для зменшення обсягу звукової, фото- й

відеоінформації і, як показує практика, для такого роду [інформації](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D1%8F) це набагато вигідніше, але чим більша втрата даних при стисненні, тим помітніші в стиснених даних стають[артефакти](https://uk.wikipedia.org/wiki/%D0%90%D1%80%D1%82%D0%B5%D1%84%D0%B0%D0%BA%D1%82%D0%B8_%D1%81%D1%82%D0%B8%D1%81%D0%BA%D1%83).

* **Допустимість втрат**[[ред.](https://uk.wikipedia.org/w/index.php?title=%D0%A1%D1%82%D0%B8%D1%81%D0%BD%D0%B5%D0%BD%D0%BD%D1%8F_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85&veaction=edit&section=3) • [ред. код](https://uk.wikipedia.org/w/index.php?title=%D0%A1%D1%82%D0%B8%D1%81%D0%BD%D0%B5%D0%BD%D0%BD%D1%8F_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85&action=edit&section=3)]
* У загальному випадку алгоритми стиснення без втрат універсальні в тому сенсі, що їх застосування безумовно можливо для даних будь-якого типу, в той час як можливість застосування стиснення зі втратами потрібно обґрунтувати.

1. Метод групового кодування

Кодирование длин серий (англ. *Run-length encoding, RLE*) или кодирование повторов — простой алгоритм сжатия данных, который оперирует *сериями данных*, то есть последовательностями, в которых один и тот же символ встречается несколько раз подряд. При кодировании строка одинаковых символов, составляющих серию, заменяется строкой, которая содержит сам повторяющийся символ и количество его повторов.

1. Словарні методи стиснення даних

Более тридцати лет алгоритм сжатия Хаффмана и его варианты оставались наиболее популярными методами. Однако в 1977 два исследователя из Израиля предложили совершенно другой подход к этой проблеме. Абрахам Лемпел и Якоб Зив выдвинули идею формирования “словаря” общих последовательностей данных. При этом сжатие данных осуществляется за счет замены записей соответствующими кодами из словаря. Так были созданы алгоритмы сжатия LZ77 и LZ78. Алгоритм LZ78 был впоследствии расширен Терри Велчем и был создан его новый вариант – LZW.

1. Метод арифметичного кодування

При арифметичному стисненні повідомлень алфавіту джерела ставиться у відповідність числовий, відкритий справа, інтервал [0,1), а кожен символ алфавіту зіставляється з різними ділянками цієї числової осі. Ширина інтервалу (*діапазон*) кожної ділянки залежить від імовірності (*частоти*) появи символу в повідомленні.

Число бітів, необхідне для подання інтервалу шириною *s*, дорівнює http://posibnyky.vntu.edu.ua/e_s/563_src/563_image005.png. Основа логарифму тут і в наступних виразах дорівнює 2. Ширина *s* останнього інтервалу рядка повідомлення, що складається з http://posibnyky.vntu.edu.ua/e_s/563_src/563_image007.pngсимволів визначається добутком ймовірностей символів http://posibnyky.vntu.edu.ua/e_s/563_src/563_image009.png повідомлення

http://posibnyky.vntu.edu.ua/e_s/563_src/563_image011.png.

В зв’язку з цим можна записати, що

http://posibnyky.vntu.edu.ua/e_s/563_src/563_image013.png,

де *m* – кількість різних символів повідомлення http://posibnyky.vntu.edu.ua/e_s/563_src/563_image015.png.

1. Факсіміальне кодування

Факс ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Fax, сокращ. от facsimile (новолат. faximile, от [лат.](https://ru.wikipedia.org/wiki/%D0%9B%D0%B0%D1%82%D0%B8%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) fac simile, сделай подобным образом , сделай подобное ), Факсимильная связь — телекоммуникационная технология передачи изображений электрическими сигналами. Исторически включалась в состав [телеграфной](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D0%B3%D1%80%D0%B0%D1%84) связи и является разновидностью [электросвязи](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D1%81%D0%B2%D1%8F%D0%B7%D1%8C)

18) **Словарные методы сжатия данных**

**Более тридцати лет алгоритм сжатия Хаффмана и его варианты оставались наиболее популярными методами. Однако в 1977 два исследователя из Израиля предложили совершенно другой подход к этой проблеме. Абрахам Лемпел и Якоб Зив выдвинули идею формирования “словаря” общих последовательностей данных. При этом сжатие данных осуществляется за счет замены записей соответствующими кодами из словаря. Так были созданы алгоритмы сжатия LZ77 и LZ78. Алгоритм LZ78 был впоследствии расширен Терри Велчем и был создан его новый вариант – LZW.**

**LZ-алгоритм**

**Этот алгоритм был впервые описан в работах Абрахама Лемпеля и Якоба Зива (двухступенчатое кодирование). На сегодняшний день этот алгоритм, известный как LZ-алгоритм, и его модификации получили наиболее широкое распространение по сравнению с другими алгоритмами сжатия. В его основе лежит идея замены наиболее часто встречающихся последовательностей символов (строк) в файле ссылками на образцы, хранящиеся в специально создаваемом словаре. Так, создав словарь, содержащий 65536 наиболее употребительных слов, можно представить текстовые файлы в виде последовательности 16-битовых ссылок на место данного слова в словаре.**

**Некоторые разновидности LZ-алгоритма**

**LZ77. Алгоритм, в котором чередуются указатели и символы. Указатели ссылаются на подстроку, расположенную в предыдущих N символах.**

**LZB. Аналогичен алгоритму LZSS, кроме кодирования указателей. LZC. Алгоритм, реализованный программой compress в UNIX-системах.**

**LZFG. Алгоритм, в котором указатели выбирают узел в дереве поиска. Строки в дереве выбираются из текущего окна.**

**LZH. Алгоритм, аналогичный LZSS, на втором этапе которого кодирование указателей осуществляется по методу Хаффмана. LZJ. В результате работы этого алгоритма генерируются только**

**указатели.Указатели означают подстроку в любом месте предыдущего текста.**

**LZMW. Алгоритм, аналогичный LZT, но фразы строятся путем объединения предыдущих фраз.**

**LZR. На выходе алгоритма чередуются указатели и символы**

**19) Архивирование данных.**

**Архивация**:

* **Архивация** — **подготовительная обработка (сбор, классификация, каталогизация, сжатие (для цифровой информации)) документов для долгосрочного хранения.**
* **Архивация файлов — перекодирование данных с целью уменьшения их объѐма.**
* [**Архивация**](http://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D0%B0%D1%80%D1%85%D0%B8%D0%B2%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) **(электронное архивирование) — запись информации в электронном виде для долговременного хранения. Не путать с созданием резервных копий данных.**

**Резервное копирование (**[**англ.**](http://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) ***backup*) — процесс создания копии данных на носителе (жѐстком диске, дискете и т. д.), предназначенном для** [**восстановления данных**](http://ru.wikipedia.org/wiki/%D0%92%D0%BE%D1%81%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) **в оригинальном месте их расположения в случае их повреждения или разрушения. Резервное копирование необходимо для возможности быстрого и недорогого** [**восстановления информации**](http://ru.wikipedia.org/wiki/%D0%92%D0%BE%D1%81%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) **(документов, программ, настроек и т. д.) в случае утери рабочей копии информации по какой-либо причине.**

[**Архив**](http://ru.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D0%B8%D0%B2) **— учреждение или структурное подразделение учреждения, организации или предприятия, осуществляющее приѐм, комплектование и хранение архивных документов в интересах пользователей.**

**Отдел учреждения, где хранятся старые документы, книги, оконченные производством дела.**

[**Архив**](http://ru.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D0%B8%D0%B2_%28%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0%29) **в информатике:**

**Файл, составленный из нескольких файлов. Для этого существуют специальные программы.**

**Файл, содержащий сжатые файлы. Для сжатия используют программы-архиваторы.**

**Архиватор — программа, осуществляющая упаковку одного и более файлов в архив или серию архивов, для удобства переноса или хранения, а также распаковку архивов. Многие архиваторы используют сжатие без потерь для уменьшения размера архива.**

**Простейшие архиваторы просто последовательно объединяют содержимое файлов в архив. Архив должен также содержать информацию об именах и длине оригинальных файлов для их восстановления. Большинство архиваторов также сохраняют метаданные файлов, предоставляемые операционной системой, такие, как время создания и права доступа.**

**Программа, создавая архив, обрабатывает как текстовые файлы, так и бинарные файлы. Первые всегда сжимаются в несколько раз (в зависимости от архиватора), тогда как сжатие бинарных файлов зависит от их характера. Одни бинарные файлы могут быть сжаты в десятки раз, сжатие же других может и вовсе не уменьшить занимаемый ими объем. Сжатие данных обычно происходит значительно медленнее, чем обратная операция.**

**Характеристики архиваторов:**

* **Степень сжатия**
* **Скорость сжатия**
* **Сервис, т.е. набор функций архиватора.**

**Характеристики архиваторов — обратно зависимые величины. То есть, чем больше скорость сжатия, тем меньше степень сжатия, и наоборот.**

**Нахождение для любого входного файла программы, сжимающей этот файл до наименьшего возможного размера, является алгоритмически неразрешимой задачей, поэтому "идеальный" архиватор невозможен.**

**20) Имеется разрыв между требованиями к верности, принимаемой данных и возможностями каналов связи. В частности, стандартами**

**международных организаций ITU-T и МОС установлено, что вероятность ошибки при телеграфной связи не должна превышать 3 x 10-5 (на знак), а при передаче данных – 10-6 (на единичный элемент, бит).**

**На практике допустимая вероятность ошибки при передаче данных может быть еще меньше – 10-9. В то же время каналы связи (особенно проводные каналы большой протяженности и радиоканалы) обеспечивают вероятность ошибки на уровне 10-3...10-4 даже при использовании фазовых корректоров, регенеративных ретрансляторов и других устройств, улучшающих качество каналов связи.**

**Кардинальным способом снижения вероятности ошибок при приеме является введение избыточности в передаваемую информацию.**

**В системах передачи информации без обратной связи данный способ реализуется в виде помехоустойчивого кодирования, многократной передачи информации или одновременной передачи информации по нескольким параллельно работающим каналам. Помехоустойчивое кодирование доступнее, при прочих равных условиях позволяет обойтись**

**меньшей избыточностью и за счет этого повысить скорость передачи информации.**

**При передаче информации по каналу связи с помехами в принятых данных могут возникать ошибки. Если такие ошибки имеют небольшую величину или возникают достаточно редко, информация может быть использована потребителем. При****большом числе ошибок полученной информацией пользоваться нельзя.**

**Ранее уже говорилось о понятии помехоустойчивости (способности информационных систем противостоять воздействию помех). Для реализации принципа помехоустойчивости информационных систем может быть использовано помехоустойчивое кодирование.**

**Помехоустойчивыми (корректирующими) называются коды, позволяющие обнаружить и при необходимости исправить ошибки в принятом сообщении.**

**Возможность использования кодирования для уменьшения числа ошибок в канале была теоретически показана К. Шенноном в 1948 году в его работе "Математическая теория связи".**

***Теория помехоустойчивого кодирования базируется на результатах исследований, проведенных* Шенноном *и сформулированных в виде теоремы:***

* **При любой производительности источника сообщений, меньшей, чем пропускная способность канала, существует такой способ кодирования, который позволяет обеспечить передачу всей информации, создаваемой источником сообщений, со сколь угодно малой вероятностью ошибки.**
* **Не существует способа кодирования, позволяющего вести передачу информации со сколь угодно малой вероятностью ошибки, если производительность источника сообщений больше пропускной способности канала.Теперь это утверждение принято именовать второй теоремой Шеннона.**

**21) ОБЩИЕ ПРИНЦИПЫ ПОМЕХОУСТОЙЧИВОГО**

**КОДИРОВАНИЯ**

**Хотя различные схемы кодирования очень непохожи друг на друга и основаны на различных *математических* теориях, всем им присущи два общих свойства.**

**Первое** − **использование избыточности**.Закодированныепоследовательности всегда содержат дополнительные, или избыточные, символы.

**Второе** — **свойство усреднения**, **означающее, что избыточные символы зависят от нескольких информационных символов, то есть информация, содержащаяся в кодовой последовательности X*,* перераспределяется также и на избыточные символы.**

**Пусть M – число знаков первичного алфавита. Длина равномерного двоичного кода K ≥ log2M – при этом каждый знак получает свою уникальную последовательность знаков вторичного алфавита. Общее число кодовых комбинаций Sр = 2K, очевидно, Sр M.**

**Например, мощность алфавита – М=40. Длина равномерного бинарного кода K  log2M = 6. С помощью 6 бит можно получить 26=64 кодовых комбинаций. Они будут считаться разрешенными, хотя некоторые из них и не соответствуют символам исходного алфавита.**

**В дальнейшем будем называть часть помехоустойчивого кода, составленную из указанных k бит, информационной (поскольку именно они содержат информацию о передаваемом знаке первичного алфавита). Если пересылать только эти информационные биты, то любое искажение, состоящее в инверсии хотя бы одного бита, приведет к появлению новой разрешенной кодовой комбинации и, следовательно, обнаружено быть не может.**

**Возможность обнаружения и исправления ошибок в помехоустойчивых кодах достигается за счет того, что после**

**первичного кодирования (установления соответствия каждому знаку первичного алфавита его кода) осуществляется вторичное**

**кодирование, в ходе которого к k информационным битам по определенным правилам добавляются r проверочных (корректирующих) бит.**

**В результате общая длина кодовой комбинации становится равной**

**n = k + r**

**В дальнейшем такие коды будем называть (n,k)-кодами, а число возможных кодовых комбинаций возрастает до S = 2n.**

**Из них не все оказываются разрешенными – их только Sр, остальные же Sr = S – Sр комбинаций являются запрещенными.**

**Например, мощность алфавита – М=40. Количество информационных бит k ≥ log2M = 6. С помощью 6 бит можно получить Sр=26=64 разрешенных кодовых комбинаций. Допустим,**

**помехоустойчивый код содержит r=3 проверочных бита. Общая длина кодового слова будет равна n = k + r = 6+3=9 бит, а общее количество кодовых комбинаций составит S=29=512. Из них разрешенных кодовых комбинаций – 64, а запрещенных – Sr = S – Sр= 512-64= 448.**

**Если при передаче возникает ошибка, она проявится в том, что разрешенная кодовая комбинация перейдет в запрещенную – это можно отследить и даже исправить. Такое обнаружение, очевидно, окажется невозможным, если в результате ошибки передачи одна разрешенная кодовая комбинация перейдет в другую разрешенную. В связи с этим возникает проблема поиска таких способов избыточного кодирования, при которых вероятность перехода одной разрешенной кодовой комбинации в другую была бы минимальной.**

**Рассмотрим влияние избыточности на корректирующие свойства кода.**

**22) Классификация помехоустойчивых кодов**

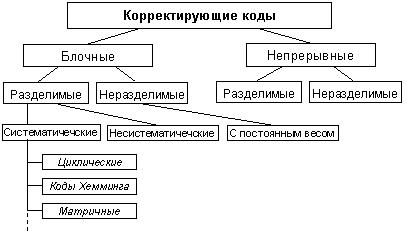
****

Рис.1. Классификация помехоустойчивых кодов.

Первый классификационный признак – коды бывают блочными или непрерывными.

**При блочном кодировании передаваемые двоичные сообщения сгруппированы в блоки, которыми кодируются знаки (или группы знаков) первичного алфавита. В блоке присутствуют информационные и проверочные биты. Известно, что если все кодовые комбинации имеют одинаковую длину, код называется равномерным; если нет – неравномерным. При декодировании удобнее (проще) иметь дело с равномерным кодом, поэтому именно он, как правило, используется в помехоустойчивом кодировании.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Непрерывные** | **(синонимы:** | **цепные,** | **сверточные,** |
| **рекуррентные)** | **кодыпредставляют** | **собой** | **непрерывную** |

**последовательность бит, не разделяемую на блоки (информационные и проверочные биты в них чередуются по определенному правилу).**

**Блочное кодирование удобно использовать в тех случаях, когда исходные данные по своей природе уже сгруппированы в какие-либо блоки или массивы.**

**При передаче по радиоканалам чаще используется сверточное кодирование, которое лучше приспособлено к побитовой передаче данных.** Кроме этого,при одинаковой избыточности сверточныекоды, как правило, обладают лучшей исправляющей способностью.

**Второй классификационные признак, относящийся как к блочным, так и к непрерывным кодам, подразделяет коды на разделимые и неразделимые.**

***Разделимыми называются коды, в которых информационные и проверочные биты располагаются в строго определенных позициях.***

**В неразделимых кодах такой определенности нет, что затрудняет их кодирование и декодирование. Поэтому практический интерес представляют в основном разделимые коды, а из неразделимых – только коды с постоянным весом.**

**Третий классификационный признак относится только к блочным разделимым кодам – они подразделяются на систематические (линейные) и несистематические.**

**Двоичный код является линейным, если сумма по модулю 2 (mod2) двух кодовых слов также является кодовым словом этого кода. В линейных кодах проверочные биты являются результатом линейных операций над информационными разрядами. В несистематических (нелинейных) кодах информационные и проверочные биты либо вообще не имеют связи, либо эта связь нелинейна – такие коды применяются крайне редко.**

***Наиболее часто в каналах связи используются блочные линейные коды, называемые (n, k)-коды, к которым относятся циклические, коды Хемминга, матричные канонические и ряд других.***

***23)* Связь корректирующей способности кода с кодовым расстоянием**

**При взаимно независимых ошибках наиболее вероятен переход в кодовую комбинацию, отличающуюся от данной в наименьшем числе символов.**

**Степень отличия любых двух кодовых комбинаций характеризуется расстоянием между ними в смысле *Хэмминга* или просто *кодовым расстоянием*.**

***Кодовое расстояние выражается числом символов, в которых комбинации отличаются одна от другой, и обозначается через d.***

**Чтобы получить кодовое расстояние между двумя комбинациями двоичного кода, достаточно подсчитать число единиц в сумме этих комбинаций по модулю 2. Например**

1001111101

* 1100001010

0101110111

**(Сложение ”по модулю 2”: y= х1  х2, сумма равна 1 тогда и только тогда, когда х1 и x2 не совпадают).**

**Минимальное** **расстояние,** **взятое** **по** **всем** **парам**

**кодовых разрешенных комбинаций кода, называют *минимальным* *кодовым расстоянием.***

**Более полное представление о свойствах кода дает *матрица* *расстояний* D,элементы которойdij(i,j = 1,2,…,m)равнырасстояниям между каждой парой из всех m разрешенных комбинаций.**

**Декодирование после приема производится таким образом, что принятая кодовая комбинация отождествляется с той разрешенной, которая находится от нее на наименьшем кодовом расстоянии.**

**Такое декодирование называется декодированием по методу максимального правдоподобия.**

**Очевидно, что при кодовом расстоянии d0=1 все кодовые комбинации являются разрешенными.**

**Например, при n=3 разрешенные комбинации образуют следующее множество: 000, 001, 010, 011, 100, 101, 110, 111.**

**Любая одиночная ошибка трансформирует данную комбинацию в другую разрешенную комбинацию. Это случай кода без избыточности, не обладающего корректирующей способностью.**

**Если d0 = 2, то ни одна из разрешенных кодовых комбинаций при одиночной ошибке не переходит в другую разрешенную комбинацию. Например, подмножество разрешенных кодовых комбинаций может быть образовано по принципу четности в нем числа единиц. Например, для n=3:**

**000, 011, 101, 110 – разрешенные комбинации; 001, 010, 100, 111 – запрещенные комбинации.**

**Код обнаруживает одиночные ошибки, а также другие ошибки нечетной кратности (при n=3 тройные).**

**В общем случае при необходимости обнаруживать ошибки кратности до r включительно минимальное расстояние по Хэммингу между разрешенными кодовыми комбинациями должно быть по крайней мере на единицу больше r, т.е**

**d0 minr+1.**

**Действительно, в этом случае ошибка, кратность которой не превышает r, не в состоянии перевести одну разрешенную кодовую комбинацию в другую.**

***Для исправления* одиночной ошибки кодовой комбинациинеобходимо сопоставить подмножество запрещенных кодовых комбинаций.**

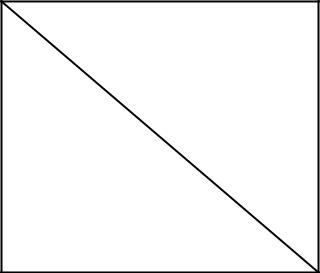
**Чтобы эти подмножества не пересекались, расстояние по Хэммингу между разрешенными кодовыми комбинациями должно быть не менее трех.**

**При n=3 за разрешенные кодовые комбинации можно, например, принять 000 и 111. Тогда разрешенной комбинации 000 необходимо приписать подмножество запрещенных кодовых комбинаций 001, 010, 100, образующихся в результате одиночной ошибки в одном из разрядов комбинации 000.**

**24) В общем случае для обеспечения возможности исправления всех ошибок кратности до s включительно при декодировании по методу максимального правдоподобия, каждая из ошибок должна приводить к запрещенной комбинации, относящейся к подмножеству исходной разрешенной кодовой комбинации.**

**Любая n-разрядная двоичная кодовая комбинация может быть интерпретирована как вершина m-мерного единичного куба, т.е. куба с длиной ребра, равной 1. При n=2 кодовые комбинации располагаются в вершинах квадрата:**

01 11

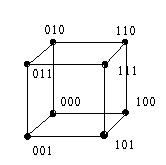


d=2

|  |  |
| --- | --- |
| 00 | 10 |

**Рис. 2.**

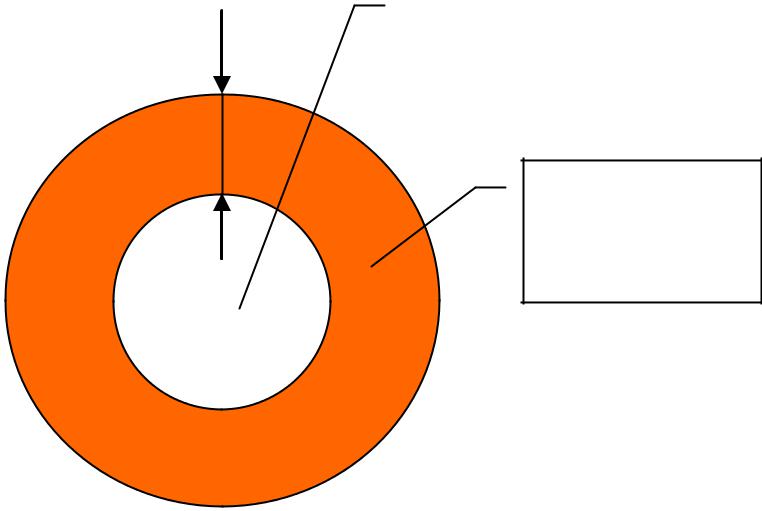
**При n=3 кодовые комбинации располагаются в вершинах единичного куба:**

****

**Рис. 3.**

**В общем случае n-мерный единичный куб имеет 2n вершин, что равно наибольшему возможному числу кодовых комбинаций.**

|  |  |
| --- | --- |
| d0 |  |
| Разрешенные |
|  |
|  | кодовые |
|  | комбинации |
|  |  |



**Запрещенные**

**кодовые**

**комбинации**

101

Разрешенные

**100 111 011** кодовые

комбинации

**110** **011** **000**

**Такая модель дает простую геометрическую интерпретацию и кодовому расстоянию между отдельными кодовыми**

**комбинациями. Оно соответствует наименьшему числу ребер единичного куба, которые необходимо пройти, чтобы попасть от одной комбинации к другой.**

**Ошибка будет не только обнаружена, но и исправлена, если искаженная комбинация остается ближе к первоначальной, чем к**

**любой другой разрешенной комбинации, то есть должно быть:**

1

* 1. **или** *d* 2*r* 1
* **общем случае для того, чтобы код позволял обнаруживать***dr*

**все ошибки кратности r и исправлять все ошибки кратности s (r>s), его кодовое расстояние должно удовлетворять неравенству**

**d  r+s+1 (rs).**

**Метод декодирования при исправлении одиночных независимых ошибок можно пояснить следующим образом. В подмножество каждой разрешенной комбинации относят все вершины, лежащие в сфере с радиусом (d-1)/2 и центром в вершине, соответствующей данной разрешенной кодовой комбинации. Если в результате действия помехи комбинация переходит в точку, находящуюся внутри сферы (d-1)/2, то такая ошибка может быть исправлена.**

**Если помеха смещает точку разрешенной комбинации на границу двух сфер (расстояние d/2) или больше (но не в точку, соответствующую другой разрешенной комбинации), то такое искажение может быть обнаружено.**

**Для кодов с независимым искажением символов лучшие корректирующие коды – это такие, у которых точки, соответствующие разрешенным кодовым комбинациям, расположены в пространстве равномерно.**

**Проиллюстрируем построение корректирующего кода на следующем примере. Пусть исходный алфавит, состоящий из четырех букв, закодирован двоичным кодом: х1 = 00; х2 = 01; х3 = 10; х4 = 11. Этот код использует все возможные комбинации длины 2, и поэтому не может обнаруживать ошибки (так как d0=1).**

**Припишем к каждой кодовой комбинации один элемент 0 или 1 так, чтобы число единиц в нем было четное, то есть х1 = 000; х2 =**

**011; х3 = 101; х4 = 110.**

**Для этого кода d0=2, и, следовательно, он способен обнаруживать все однократные ошибки. Так как любая запрещенная комбинация содержит нечетное число единиц, то для обнаружения ошибки достаточно проверить комбинацию на четность (например, суммированием по модулю 2 цифр кодовой комбинации). Если число единиц в слове четное, то сумма по модулю 2 его разрядов будет 0, если нечетное – то 1. Признаком четности называют инверсию этой суммы.**

**Рассмотрим общую схему организации контроля по четности (контроль по нечетности, parity check – контроль по паритету).**

**25) В случае поблочного кодирования, где каждый из блоков состоит из *М* независимых букв *,* минимальная средняя *длина кодового блока лежит в пределах***

******

**(50)**

**Общее выражение среднего числа элементарных символов на букву сообщения пи блочном кодировании**

**0,81**

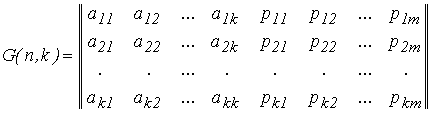
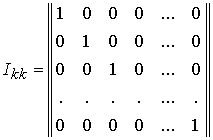
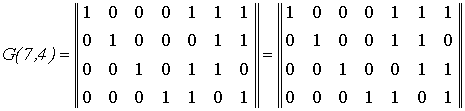
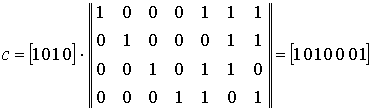
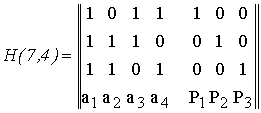
**0,09**

**0,09**

**0,01**

**С точки зрения информационной нагрузки на символ сообщения поблочное кодирование всегда выгоднее, чем побуквенное.**

**Суть блочного кодирования можно уяснить на примере представления десятичных цифр в двоичном коде. Так, при передаче числа 9 в двоичном коде необходимо затратить 4 символа, т. е. 1001. Для передачи числа 99 при побуквенном кодировании - 8, при поблочном - 7, так как 7 двоичных знаков достаточно для передачи любого числа от 0 до 127; при передаче числа 999 соотношение будет 12 - 10, при передаче числа 9999 соотношение будет 16 - 13 и т. д. В общем случае «выгода» блочного кодирования получается и за счет того, что в блоках происходит выравнивание вероятностей отдельных символов, что ведет к повышению информационной нагрузки на символ.**

**26-28) *Лінійним*** називається код, в якому перевірочні [символи](http://ua-referat.com/%D0%A1%D0%B8%D0%BC%D0%B2%D0%BE%D0%BB) являють собою лінійні комбінації інформаційних. ***Груповим*** називається код, який утворює алгебраїчну групу по відношенню операції додавання за модулем два.   
**Властивість лінійного коду:** сума (різниця) кодових [векторів](http://ua-referat.com/%D0%92%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D1%8B) линів-ного коду дає вектор, що належить цим кодом. **Властивість групового коду:** мінімальне кодова відстань між кодовими векторами одно мінімальній вазі ненульових векторів. Вага кодового вектора дорівнює числу одиниць в кодової комбінації.   
Групові коди зручно задавати за допомогою матриць, розмірність яких визначається параметрами *k* і *n.* Число рядків дорівнює *k,* а число стовпців одно *n = k + m.*   
 . (6)   
  
Коди, породжувані цими матрицями, називаються *(n, k)-кодами,* а [відповідні](http://ua-referat.com/%D0%92%D1%96%D0%B4%D0%BF%D0%BE%D0%B2%D1%96%D0%B4%D1%8C) їм [матриці](http://ua-referat.com/%D0%9C%D0%B0%D1%82%D1%80%D0%B8%D1%86%D1%96) породжують (утворюючими, що виробляють). Породжує матриця *G* складається з [інформаційної](http://ua-referat.com/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D1%8F) *I kk* і перевірочної *R km* матриць. Вона є стислим описом лінійного коду і може бути представлена ​​в канонічній (типової) форми   
http://ua-referat.com/dopb317838.zip . (7)   
В якості інформаційної матриці зручно використовувати одиничну матрицю, ранг якої визначається кількістю інформаційних розрядів   
 . (8)   
Рядки одиничної матриці представляють собою лінійно-незалежні комбінації (базисні вектора), тобто їх по парне підсумовування за модулем два не призводить до нульової рядку.   
Рядки породжує матриці являють собою перші *k* комбінацій коригуючого коду, а інші кодові комбінації можуть бути отримані в результаті підсумовування за модулем два всіляких поєднань цих рядків.   
Стовпці додаткової матриці *R km* визначають правила формування перевірок. Кількість одиниць в кожному рядку додаткової матриці повинна задовольняти умові *r 1* *³ d 0 -1,* але число одиниць визначає число суматорів за модулем 2 в шифратора і дешифратор, і чим їх більше, тим складніше апаратура.   
Твірна матриця коду *G (7,4)* може [мати](http://ua-referat.com/%D0%9C%D0%B0%D1%82%D0%B8) вигляд   
 і т.д.   
[Процес](http://ua-referat.com/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81) кодування полягає у взаємно - однозначним дотриманням *k-розрядних* інформаційних слів - *I* і *n-розрядних* кодових слів - *з*   
*c =****IG.*** (9)   
Наприклад: інформаційного слову *I* = [1 0 1 0] відповідає наступне кодове [слово](http://ua-referat.com/%D0%A1%D0%BB%D0%BE%D0%B2%D0%BE)   
 . (10)   
При цьому, [інформаційна](http://ua-referat.com/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D1%8F) частина залишається без змін, а коригувальні розряди визначаються шляхом підсумовування за модулем два тих рядків перевірочної матриці, номери яких співпадають з номерами розрядів, що містять одиницю в [інформаційній](http://ua-referat.com/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D1%8F) частини коду.   
[Процес](http://ua-referat.com/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81) декодування полягає у визначенні відповідності прийнятого кодового слова, переданому інформаційного. Це здійснюється за допомогою перевірочної матриці *H (n, k).*   
http://ua-referat.com/dopb317842.zip , (11)   
  
де *R mk T-транспонована* перевірочна матриця (поміняти рядка на стовпці); *I mm* - Одинична матриця.   
Для (7, 4) - коду перевірочна матриця має вигляд   
 . (12)   
Тим *G (n, k)* і *H (n, k)* існує однозначний зв'язок, тому що вони визначаються [відповідно](http://ua-referat.com/%D0%92%D1%96%D0%B4%D0%BF%D0%BE%D0%B2%D1%96%D0%B4%D1%8C) до правил перевірки, при цьому для будь-якого кодового слова має виконуватися рівність *cH T = 0.*   
Рядки перевірочної матриці визначають правила формування перевірок. Для (7, 4)-коду   
*p 1 Å + a 1 Å + a 2 Å a 4 = S 1;*   
*p 2 Å + a 1 Å + a 2 Å a 3 = S 2;* (13)   
*p 3 Å + a 1 Å + a 3 Å a 4 = S 3.*   
Отриманий синдром порівнюємо зі стовпцями матриці і визначаємо розряд, в якому сталася помилка, номер стовпця дорівнює номером помилкового розряду. Для виправлення помилки помилковий біт необхідно проінвертіровать.

29) **Код с проверкой на четность.**

**Самым простым линейным блочным кодом является (n,n-1)-код, построенный с помощью одной общей проверки на четность. Например, кодовое слово (4,3)-кода можно записать в виде:**

**U = (m0, m1, m2, m0  m1  m2),**

**где m*i* - символы исходной информационной последовательности, принимающие значения 0 и 1, а суммирование производится по модулю 2 и обозначается символом . Результат суммирования x  y = z определяется в соответствии со следующей таблицей**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **x** | **0** | **1** | **0** | **1** |
|  |  |  |  |  |
| **y** | **0** | **0** | **1** | **1** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| **z** | **0** | **1** | **1** | **0** |
|  |  |  |  |  |

Основная идея проверки на четность состоит в следующем. Пусть информационная последовательность источника имеет вид

**m = (1 0 1)**.

Тогда соответствующая ей кодовая последовательность будет выглядеть так:

**U = (u0, u1, u2, u3) = (1 0 1 0),**

где проверочный символ **u3** формируется путем суммирования символов информационной последовательности **m**:

**u3 = m0 m1 m2** = **1  0  1 = 0**.

Если число единиц в последовательности **m** четно, то результатом суммирования будет **0**, если нечетно — **1**, то есть проверочный символ дополняет кодовую последовательность таким образом, чтобы количество единиц в ней было четным.

При передаче по каналам связи в принятой последовательности возможно появление ошибок, то есть символы принятой последовательности могут отличаться от соответствующих символов

переданной кодовой последовательности (нуль переходит в единицу, а

**1** −в **0**).

Если при передаче рассматриваемого (**4,3**)-кода произошла одна ошибка, причем неважно, в какой его позиции, то общее число единиц в принятой последовательности уже не будет четным.

Таким образом, признаком отсутствия ошибки в принятой последовательности может служить четность числа единиц. Поэтому такие коды и называются кодами с проверкой на четность.

Правда, если в принятой последовательности произошло две ошибки, то общее число единиц в ней снова станет четным и ошибка обнаружена не будет. При независимых ошибках вероятность двойной ошибки значительно меньше вероятности одиночной, поэтому наиболее вероятные одиночные ошибки таким кодом обнаруживаться все же будут.

Отметим следующий момент. Если посимвольно сложить два кодовых слова, принадлежащих рассматриваемому (**4, 3**)-коду:

**a = (a0, a1, a2, a0  a1  a2)**,и **b = (b0, b1, b2, b0  b1  b2)**,

то получим

* + **= (a0  b0, a1  b1, a2  b2, a0  b0  a1  b1  a2  b2) = (c0, c1, c2, c0**
* **c1  c2)**,

то есть **проверочный символ в новом слове с определяется по** **тому же правилу, что и в слагаемых. Поэтому с также является кодовым словом данного кода.**

**Этот пример отражает важное свойство линейных блочных кодов — замкнутость, означающее, что сумма двух кодовых слов данного кода также является кодовым словом.**

**Несмотря на свою простоту и не очень высокую эффективность, коды с проверкой на четность широко используются в системах передачи и хранения информации.** Они ценятся за невысокую избыточность:достаточнодобавить к передаваемой последовательности всего один избыточный символ − и можно узнать, есть ли в принятой последовательности ошибка. Правда, определить место этой ошибки и, следовательно, исправить ее, пока нельзя. Можно лишь повторить передачу слова, в котором была допущена ошибка, и тем самым ее исправить.

30) **Итеративный код. Еще одна простая схема кодирования, которая также часто используется, может быть построена следующим образом.**

**Предположим, что нужно передать, к примеру, девять информационных символов m = (m1 , m2 , ..., m9)*.* Эти символы можно расположить в виде квадратной матрицы, как это показано в таблице, и добавить к каждой строке и каждому столбцу этой таблицы по проверочному символу (проверка на четность).**

|  |  |  |  |
| --- | --- | --- | --- |
| **m1** | **m2** | **m3** | **m1  m2  m3** |
| **m4** | **m5** | **m6** | **m4  m5  m6** |
| **m7** | **m8** | **m9** | **m7  m8  m9** |
| **m1  m4  m7** | **m2  m5  m8** | **m3  m6  m9** | **m1  m2  m3  …  m9** |

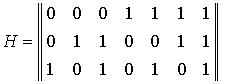
**Таким образом, по строкам и по столбцам этой таблицы будет выполняться правило четности единиц.**

**Если в процессе передачи по каналу с помехами в этой таблице произойдет одна ошибка (например, в символе m4), то проверка на четность в соответствующей строке и столбце не будет выполняться. Иными словами, координаты ошибки однозначно определяются номерами столбца и строки, в которых не выполняются проверки на четность. Таким образом, этот код, используя различные проверки на четность (по строкам и по столбцам), способен не только обнаруживать, но и исправлять ошибки (если известны координаты ошибки, то ее исправление состоит просто в замене символа на противоположный: если 0, то на 1, если 1 – то на 0).**

**Описанный метод кодирования, называемый итеративным, оказывается полезным в случае, когда данные естественным образом формируются в виде массивов, например, на шинах ЭВМ, в памяти, имеющей табличную структуру, и т.д. При этом размер таблицы в принципе не имеет значения (3×3 или 20×20), однако в первом случае будет исправляться одна ошибка на 3×3=9 символов, а во втором – одна на 20×20=400 символов.**

**Если в простом коде с проверкой на четность для обнаружения ошибки приходится добавлять к информационной последовательности всего один символ, то для того, чтобы код стал исправлять однократную ошибку, понадобилось к девяти**

**информационным символам добавить еще семь проверочных. Таким образом, избыточность этого кода оказалась очень большой, а исправляющая способность – сравнительно низкой. Поэтому усилия специалистов в области помехоустойчивого кодирования всегда были направлены на поиск таких кодов и методов кодирования, которые при минимальной избыточности обеспечивали бы высокую исправляющую способность.**

**31) Код Хеммінга**   
Код Хеммінга відноситься до класу лінійних кодів і являє собою систематичний код - код, в якому [інформаційні](http://ua-referat.com/%D0%86%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D1%96%D1%8F) та [контрольні](http://ua-referat.com/%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C) біти розташовані на суворо визначених місцях в кодової комбінації.   
Код Хеммінга, як і будь-який *(n, k)* - код, містить *до* інформаційних і *m = nk* надлишкових (перевірочних) біт.   
Надмірна частина коду будується т. о. щоб можна було при декодуванні не тільки [встановити](http://ua-referat.com/%D0%92%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%B8) наявність помилки але, і вказати номер позиції в якій сталася помилка, а значить і виправити її, Інвертувати значення відповідного біта.   
Існують різні методи реалізації коду Хеммінга і кодів які є модифікацією коду Хеммінга. Розглянемо алгоритм побудови коду для виправлення одиночної помилки.   
1. По заданій кількості інформаційних символів - *k,* або інформаційних комбінацій *N = 2 k,* використовуючи співвідношення:   
*n = k + m, 2 n* *³* *(n +1) 2 k* і *2 m ³ n +1* (14)   
*m = [log 2 {(k +1) + [log 2 (k +1)]}]*   
обчислюють основні параметри коду *n* і *m.*   
2. Визначаємо робітники і контрольні позиції кодової комбінації. Номери контрольних позицій визначаються за законом *2 i,* де *i* = 1, 2, 3, ... тобто вони рівні 1, 2, 4, 8, 16, ... а інші позиції є робочими.   
3. Визначаємо значення контрольних розрядів (0 або 1) за допомогою багаторазових перевірок кодової комбінації на парність. Кількість перевірок одно *m = nk.*У кожну перевірку включається один контро-ний і певні перевірочні біти. Якщо результат перевірки дає парне число, то [контрольному](http://ua-referat.com/%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C) біту присвоюється значення -0, в іншому випадку - 1. Номери інформаційних біт, що включаються в кожну перевірку, визначаються за допомогою бінарного коду натуральних *n-чисел* розрядністю - *m* (табл. 1, для *m =* 4) або за допомогою перевірочної матриці *H (m 'n),* стовпці якої представляють запис у двійковій системі всіх цілих чисел від 1 до *2 k -* 1 перерахованих у зростаючому порядку. Для *m =* 3 перевірочна матриця має вигляд   
 . (15)   
  
Кількість розрядів *m* - визначає кількість перевірок.   
У першу перевірку включають коефіцієнти, що містять 1 в молодшому (першому) розряді, тобто b 1, b 3, b 5 і т. д.   
 У другу перевірку включають коефіцієнти, що містять 1 в другому розряді, тобто b 2, b 3, b 6 і т. д.   
 У третю перевірку - коефіцієнти які містять 1 у третьому розряді і т. д.   
Таблиця 1

|  |  |  |  |
| --- | --- | --- | --- |
| Десяткові числа  (Номери розрядів  кодової комбінації) | Двійкові числа і їх розряди | | |
| 3 | 2 | 1 |
| 1  2  3  4  5  6  7 | 0  0  0  1  1  1  1 | 0  1  1  0  0  1  1 | 1  0  1  0  1  0  1 |

Для виявлення та виправлення помилки складаються аналогічні перевірки на парність контрольних сум, результатом яких є двійкове *(nk)-розрядне* число, зване синдромом і вказує на становище помилки, тобто номер помилкову позицію, який визначається за двійковій запису числа, або за перевірочної матриці.   
Для виправлення помилки необхідно проінвертіровать біт у помилкову позицію. Для виправлення одиночної помилки і виявлення подвійної використовують додаткову перевірку на парність. Якщо при виправленні помилки [контроль](http://ua-referat.com/%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C) на парність фіксує помилку, то значить в кодової комбінації дві помилки.   
Схема кодера і декодера для коду Хеммінга наведена на рис. 1.

32) Основи поліноміальних кодів

Представлення кодового слова (n, k)-коду у вигляді послідовності U = (U0, U1, ..., Un-1 ) довжиною n символів або їх завдання за допомогою системи перевірочних рівнянь і породжує матриці не є єдино можливим. Ще один зручний і широко використовуваний спосіб представлення того ж кодового слова полягає в тому, що елементи U0, U1, ..., Un-1 є коефіцієнтами многочлена від X, тобто

(х) = f(х) = U0 +U1\* Х + U2\*Х2 +...+Un- 1\*Хn-1 . (1.51)

Використовуючи це подання, можна визначити поліноміальний код як безліч всіх многочленів ступеня, не більшої n -1, містять в якості загального множника деякий фіксований многочлен g(x).

Многочлен g(x) називається породжується многочленом коду. Представлення кодових слів у такій формі дозволяє звести дії над комбінаціями символів до дії над поліномами. Визначимо дії над поліномами в полі двійкових символів GF(2).Сумою двох поліномів f(x) і g(x) з GF(2)називається поліном з GF (2), який визначається наступним чином:

f(x)\* g(x)= (1.52)

Іншими словами, додаванню двійкових поліномів відповідає додавання по mod2 коефіцієнтів при однакових степенях х.

Наприклад:

Твором двох поліномів з GF (2) називається поліном з GF (2), який визначається наступним чином:

(x)\* g(x)=

тобто твір виходить за звичайним правилом перемножування статичних функцій, проте одержувані коефіцієнти при даному степенні Х складаються по модулю 2.

Наприклад:

Нарешті, можна сформулювати теорему про поділ поліномів: для кожної пари поліномів С(х) и d(x), d(x) ¹ 0 0 існує єдина пара поліномів q(x) - частка і p(х) - залишок, такі, що

С(х) = q(x)\* d(x) + p(х) , (1.58)

де степінь залишку p(х) менше степеня дільника d(x).

Іншими словами, поділ поліномів проводиться за правилами ділення статичних функцій, при цьому операція віднімання замінюється підсумовуванням за mod2.

Наприклад:

Ще раз нагадаємо, що при додаванні по mod2 сума двох одиниць (тобто двох елементів полінома з однаковими степенями) дорівнюватиме нулю, а не звичним у десятковій системі числення двом. І, крім цього, операції вирахування і складання по mod2 збігаються.

33) Кодування з використанням циклічних кодів

Припустимо, треба закодувати деяку інформаційну послідовність

m = (m0, m1, m2 … mk-1), (1. 62)

Відповідний їй поліном виглядає наступним чином:

m (x) = m0 + m1 \* x + m2 \* x2 + … + mk-1 \* xk-1. (1. 63)

Помноживши m (x) на xn-k:

xn-k \* m (x) = m0 \* xn-k + m1 \* xn-k +1 + … + mk-1 \* xn-1, (1. 64)

отримаємо поліном степеня n-1 або меншою.

Скориставшись теоремою про поділ поліномів, можна записати

xn-k \* m (x) = q (x) \* g (x) + p (x), (1. 65)

де q (x) і p (x) -частка і залишок від ділення полінома xn-k \* m (x)

а породжує поліном g (x).

Оскільки степінь g (x) дорівнює (n-k), то степінь p (x) повинна бути (nk-1) або менше, а сам поліном p (x) буде мати вигляд

p (x) = p 0 + p 1 \* x + p 2 \* x2 + … + p n-k-1 \* xn-k-1, (1. 66)

З урахуванням правил арифметики в GF (2) даний вираз можна переписати таким чином:

p (x) + xn-k \* m (x) = q (x) \* g (x), (1. 67)

звідки видно, що поліном p (x) + xn-k \* m (x) є кратним g (x) і має степінь n-1 або меншу. Отже, поліном p (x) + xn-k \* m (x) — це кодовий поліном, відповідний до кодованої інформаційної послідовності m (x).

|  |
| --- |
|  |
| U = | (0, 1 … n- k-1 , | m0, m1 … mk-1), |  |
|  | ***Перевірочні символи*** | Інформаційні символи |  |
|  |  |  |  |

Розкривши останній вираз, отримаємо

p (x) + m (x) \* xn-k = p 0 + p 1 x + p 2×2. + p n-k-1 xn-k-1 + m0 xn-k + m1 \* xn-k + 1 +. + mk-1 xn-1,

що відповідає кодовому слову

U = (p 0, p 1 … p n-k-1, m0, m1 … mk-1),

перевірочні символи інформаційні символи

Таким чином, кодове слово складається з незмінною інформаційної частини m, перед якою розташовано (n-k) перевірочних символів. Перевірочні символи є коефіцієнтами полінома p (x), тобто залишку від ділення m (x) \* xn-k на породжує поліном g (x).

Щоб отриманий результат був зрозуміліше, нагадаємо, що множенню деякого двійкового полінома на xn-k відповідає зсув двійковій послідовності m = (m0, m1 … mk-1) на n-k розрядів вправо.

Розглянемо приклад. З використанням коду, що задається породжує поліномом g (x) = 1 + x + x3, закодуємо довільну послідовність, наприклад m = (0111).

Послідовності m = (0111) відповідає поліном m (x) = x + x2 + x3.

Помножимо m (x) на xn-k:

m (x) \* xn-k = m (x) \* x3 = (x + x2 + x3) \* x3 = x4 + x5 + x6, (1. 68)

Розділимо m (x) \* xn-k на породжує поліном g (x):

Таким чином, кодовий поліном, відповідний інформаційної послідовності m = (0111), буде мати наступний вигляд:

U (x) = 0X0 + 0X1 + 1X2 + 0X3 + 1X4 + 1X5 + 1X6, (1. 70)

а відповідне кодове слово U = (10 111).

Отже, циклічний (n, k)-код k-розрядної інформаційної послідовності m = (m0, m1 … mk-1) отримують таким чином:

— Інформаційну послідовність m множать на xn-k, тобто зрушують вправо на n-k розрядів;

— Поліном отриманої послідовності ділять на породжує поліном коду g (x);

— Отриманий залишок від ділення m (x) \* xn-k на g (x) додають до m (x) \* xn-k, тобто записують у молодших n-k розрядах коду.

Алгоритм кодування, заснований на розподілі поліномів, можна реалізувати, використовуючи схему поділу. Вона являє собою регістр зсуву, в якому ланцюга зворотного зв’язку замкнуті відповідно c коефіцієнтами породжує полінома g (x)

34) не знаююююююююююююююююю

Может код Хеминга? ☺)))))))))))))))))

**Указатели означают строку, расположенную в любом месте предыдущего текста.**

**LZSS. Алгоритм, в котором указатели и символы разделяются битом флага. Указатели ссылаются на подстроки в предыдущих N символах. LZT. Аналогичен LZC, но с усовершенствованным обновлением словаря.**

**LZW. На выходе этого алгоритма имеются только указатели фиксированного объема, ссылающиеся на предварительно выделенные подстроки.**

11

**Например, если программа сжатия уже имеет в словаре последовательность "АБВ" и обнаружит последовательность "АБВА", то она вначале занесет в выходной файл код из словаря для "АБВ", затем для символа "A", после чего последовательность "АБВА" будет добавлена в словарь. Если же она позже встретит "АБВАБ", то выведет код для "АБВА", затем код для символа "Б" и добавит "АБВАБ" в словарь. Когда программа встречает последовательность из словаря, она выдает код и добавляет новую запись, которая на один байт длиннее. То есть каждый раз при повторении последовательности словарь будет расти за счет включения продолжения этой последовательности.**

***На практике приходится оперировать таблицами, заполняемыми по мере сканирования файла.* При этом уже просмотренная частьфайла используется как словарь. Алгоритм основывается на движении по потоку данных скользящего окна, состоящего из двух частей: большей по объему, в которой содержатся уже обработанные данные, и меньшей, в которую по мере просмотра помещается вновь считанная информация. Во время считывания каждой новой порции данных происходит проверка, и если оказывается, что такая строка уже есть в словаре, то она заменяется ссылкой. Существует довольно большое семейство LZ-подобных алгоритмов, различающихся, например, методом поиска повторяющихся цепочек.**

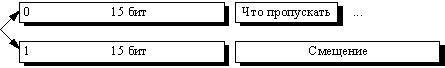
**Один из достаточно простых вариантов** LZ **алгоритма, например,**

**предполагает, что во входном потоке идет либо пара <счетчик, смещение относительно текущей позиции>, либо просто <счетчик> “пропускаемых” байт и сами значения байтов (как во втором варианте алгоритма RLE). При распаковке для пары <счетчик, смещение> копируются <счетчик> байт из выходного массива, полученного в результате разархивации, на <смещение> байт раньше, а <счетчик> (т.е. число равное счетчику) значений “пропускаемых” байт просто копируются в выходной массив из входного потока.**

**Данный алгоритм является несимметричным по времени, поскольку требует полного перебора буфера при поиске одинаковых подстрок. В результате нам сложно задать большой буфер из-за резкого возрастания времени компрессии. Однако потенциально построение алгоритма, в котором на <счетчик> и на <смещение> будет**

12

**выделено по 2 байта (старший бит старшего байта счетчика — признак повтора строки / копирования потока), даст нам возможность сжимать все повторяющиеся подстроки размером до 32Кб в буфере размером 64Кб.**

****

**При этом мы получим увеличение размера файла в худшем случае на 32770/32768 (в двух байтах записано, что нужно переписать в выходной поток следующие 215 байт), что совсем неплохо. Максимальный коэффициент сжатия составит в пределе 8192 раза. В пределе, поскольку максимальное сжатие мы получаем, превращая 32Кб буфера в 4 байта, а буфер такого размера мы накопим не сразу. Однако, минимальная подстрока, для которой нам выгодно проводить сжатие, должна состоять в общем случае минимум из 5 байт, что и определяет малую ценность данного алгоритма. К достоинствам LZ можно отнести чрезвычайную простоту алгоритма декомпрессии.**