Кодирование текста методом Хэмминга

Коды Хэмминга — наиболее известные самоконтролирующиеся (позволяющие обнаружить наличие ошибок) и самокорректирующиеся (позволяющие обнаружить непосредственное место ошибки) коды, построение которых происходит применительно к двоичной системе счисления. В соответствии с методом Хэмминга к исходному сообщению добавляют контрольные биты, каждый из которых проверяет наличие ошибки в определенной группе битов сообщения. Другими словами, это алгоритм, который позволяет закодировать какое-либо информационное сообщение определённым образом и после передачи (например по сети) определить появилась ли какая-то ошибка в этом сообщении (к примеру из-за помех) и, при возможности, восстановить это сообщение.

Код Хэмминга состоит из двух частей. Первая часть кодирует исходное сообщение, вставляя в него в определённых местах контрольные биты (вычисленные особым образом). Вторая часть получает входящее сообщение и заново вычисляет контрольные биты (по тому же алгоритму, что и первая часть). Если все вновь вычисленные контрольные биты совпадают с полученными, то сообщение получено без ошибок.

Для того, чтобы понять работу данного алгоритма, рассмотрим пример:

1. Подготовка

Допустим, у нас есть сообщение «ЯГТУ», которое необходимо передать без ошибок. Для этого сначала нужно закодировать сообщение при помощи Кода Хэмминга, представив его в бинарном виде (перевести десятичное представление символа в двоичный вид)

ASCII таблица кодов символов Windows (Win-1251)

Dec	**Символ**	**Dec**	**Символ**
032	сцеп. SP (Пробел)	224	а
044	,	225	б
046	.	226	в
192	А	227	г
193	Б	228	д
194	В	229	е
195	Г	230	ж
196	Д	231	з
197	Е	232	и
198	Ж	233	й
199	З	234	к
200	И	235	л
201	Й	236	м
202	К	237	н

203	Л	238	о
204	М	239	п
205	Н	240	р
206	О	241	с
207	П	242	т
208	Р	243	у
209	С	244	ф
210	Т	245	х
211	У	246	ц
212	Ф	247	ч
213	Х	248	ш
214	Ц	249	щ
215	Ч	250	ъ
216	Ш	251	ы
217	Щ	252	ь
218	Ъ	253	э
219	Ы	254	ю
220	Ь	255	я
221	Э		
222	Ю		
223	Я		

Таблица кодов символов для нашего сообщения

Символ	**ASCII код (Dec)**	**Двоичное представление**
Я	223	11011111
Г	195	11000011
Т	210	11010010
У	211	11010011

На этом этапе стоит определиться с, так называемой, длиной информационного слова, то есть длиной строки из нулей и единиц, которые мы будем кодировать. Допустим, у нас длина слова будет равна 16. Таким образом, нам необходимо разделить наше исходное сообщение («ЯГТУ») на блоки по 16 бит, которые мы будем потом кодировать отдельно друг от друга. Так как один символ занимает в памяти 8 бит, то в одно кодируемое слово помещается ровно два ASCII символа. Итак, мы получили две бинарные строки по 16 бит:

Я	Г		Т	У
1101111111000011			1101001011010011	

После этого процесс кодирования распараллеливается, и две части сообщения («ЯГ» и «ТУ») кодируются независимо друг от друга.

Рассмотрим, как это делается на примере первой части. Прежде всего, необходимо вставить контрольные биты. Они вставляются в строго определённых местах — это позиции с номерами, равными степеням двойки. В нашем случае (при длине информационного слова в 16 бит) это будут позиции 1, 2, 4, 8, 16. Соответственно, у нас получилось 5 контрольных бит (выделены красным цветом):

Я	Г	⇨	Я	Г
1101111111000011			00101010111111000011	

Таким образом, длина всего сообщения увеличилась на 5 бит. До вычисления самих контрольных бит, мы присвоили им значение «0».

2. Вычисление контрольных бит.

Теперь необходимо вычислить значение каждого контрольного бита. Значение каждого контрольного бита зависит от значений информационных бит, но не от всех, а только от тех, которые этот контрольных бит контролирует. Для того, чтобы понять, за какие биты отвечает каждых контрольный бит необходимо понять закономерность: контрольный бит с номером N контролирует все последующие N бит через каждые N бит, начиная с позиции N. Рассмотрим на иллюстрации:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
0	0	1	0	1	0	1	0	1	1	1	1	1	1	0	0	0	0	0	1	1	
x		x		x		x		x		x		x		x		x		x		x	1
	x	x			x	x			x	x			x	x			x	x			2
			x	x	x	x					x	x	x	x					x	x	4
							x	x	x	x	x	x	x	x							8
															x	x	x	x	x	x	16

Здесь знаком «X» обозначены те биты, которые контролирует контрольный бит, номер которого справа. То есть, к примеру, бит номер 12 контролируется битами с номерами 4 и 8. Таким образом, чтобы узнать какими битами контролируется бит с номером N надо просто разложить N по степеням двойки.

Для вычисления значений каждого контрольного бита берём каждый контрольный бит и смотрим сколько среди контролируемых им битов единиц, получаем некоторое целое число и, если оно чётное, то ставим ноль, в противном случае ставим единицу. Можно наоборот, если число чётное, то ставим единицу, в противном случае, ставим 0. Главное, чтобы в «кодирующей» и «декодирующей» частях алгоритм был одинаков.

Высчитав контрольные биты для нашего информационного слова первым способом получаем следующее:

Я	Г
11111010111111000011	

и для второй части:

Т	У
10101010010110110011	

3. Декодирование и исправление ошибок.

Теперь, допустим, мы получили закодированную первую часть сообщение, но оно пришло к нам с ошибкой. К примеру, мы получили следующую последовательность (12-ый бит передался неправильно):

Я	Г
11111010111011000011	

Эта часть алгоритма заключается в том, что необходимо заново вычислить все контрольные биты (так же, как и в первой части) и сравнить их с контрольными битами, которые мы получили. Так, посчитав контрольные биты с неправильным 12-ым битом мы получаем следующее:

Я	Г
11101011111011000011	

Как мы видим, контрольные биты под номерами: 4 и 8 не совпадают с такими же контрольными битами, которые мы получили. Теперь просто сложив номера позиций неправильных контрольных бит (4 + 8 = 12) мы получаем позицию ошибочного бита. Теперь инвертировав его и отбросив контрольные биты, получим исходное сообщение в первозданном виде.

Длинаинформационного сообщения была выбрана 16 бит, так как она наиболее оптимальная для рассмотрения примера (не слишком длинная и не слишком короткая), но конечно же длину можно взять любую. Только стоит учитывать, что в данной простой версии алгоритма на одно информационное слово можно исправить только одну ошибку.

Задание:Выполнить рассмотренный алгоритм для своего полного имени.