2) расчетный коэффициент эффективности капитальных затрат;
3) срок окупаемости информационно-аналитической системы;
4) показатель снижения стоимостных затрат за год;
5) показатель снижения трудовых затрат за год.

УДК 004(063)

Модификация алгоритма Витерби

А.Ю. Артамонов, А.Н. Гамова
СГУ, г. Саратов

Первым сверточный код предложил в 1955 г. наш соотечественник Лев Матвеевич Финк. Он предлагал в поток информационных символов включать корректирующие символы, так что между двумя информационными символами был бы корректирующий:

$$a_1 b_1\, a_2 b_2\, a_3 b_3\, \ldots\ldots a_k b_k\, a_{k+1} b_{k+1}\, \ldots.$$

Информационные символы определяются передаваемым сообщением, а корректирующие формируются по следующему правилу:

$$b_i = a_{k-S} + a_{k+S+1} \pmod 2, \qquad (1.1)$$

где s – произвольное целое число, называемое шагом кода ($s = 0,1,2\ldots$).

В принятой последовательности для каждого b_k проверяется соотношение (1.1). Если оно оказалось не выполненным при двух значениях k ($k = k_1$ и $k = k_2$) и при этом $k_2 - k_1 = 2s+1$, то информативный элемент ak_{1+S+1} должен быть заменен на противоположный. Избыточность такого кода ½. Он позволяет исправлять большинство ошибочно принятых символов, кроме некоторых неудачных сочетаний.

На стороне приема осуществляется та же самая процедура получения проверочных символов, что и на стороне передачи (1.1), и производится сравнение их с принятыми проверочными символами. Если при приеме ошибок нет, то результат суммирования по модулю 2 (сравнение) будет состоять из последовательности, содержащей одни нули. Эта последовательность, так же как в блочных циклических кодах, называется синдромом. Для возможности исправления пачек ошибок в коде Финка применяется отличный от нуля шаг s.

Сверточный кодер – это устройство, воспринимающее за каждый такт работы в общем случае k входных информационных символов и выдающее на выход за тот же такт n выходных символов, подлежащих передаче по каналу связи. На практике, как правило, используются

двоичные сверточные коды. Отношение $R = k/n$ называют относительной скоростью кода.

Основными элементами сверточного кодера являются: регистр сдвига, сумматоры по модулю 2 и коммутатор. Регистр сдвига является динамическим запоминающим устройством, в котором хранятся символы 0 или 1. Число триггерных ячеек m в регистре сдвига определяет память кода. В момент поступления на вход регистра нового информационного символа символ, хранящийся в крайнем правом разряде, выводится из регистра и сбрасывается. Каждый из остальных хранящихся в регистре символов перемещается на один разряд вправо, освобождая тем самым крайний левый разряд, куда поступает новый информационный символ.

Сумматор по модулю 2 осуществляет сложение поступающих на его входы символов 0 и 1.

Коммутатор осуществляет последовательное считывание поступающих на его входы (контакты) символов и устанавливает на выходе очередность посылки кодовых символов в канал связи.

Одним из способов задания сверточных кодов является полиномиальный способ. Сверточный кодер можно представить в виде набора из n полиномиальных генераторов, по одному для каждого из n сумматоров по модулю 2. Коэффициенты возле каждого слагаемого полинома порядка (m - 1) равны либо 1, либо 0.

Развитие теории сверточных кодов происходило в трех направления, в соответствии с тремя важнейшими методами декодирования сверточных кодов: метод порогового декодирования, декодирование по максимуму правдоподобия и метода последовательного декодирования [1].

Декодирование по методу максимума правдоподобия. Если все входные последовательности сообщений равновероятны, минимальная вероятность ошибки получается при использовании декодера, который сравнивает условные вероятности и выбирает максимальную. Условные вероятности также называют функциями правдоподобия $P\left(Z \mid B^{(m)}\right)$, где Z — это принятая последовательность, а $B^{(m)}$ — одна из возможных переданных последовательностей. Декодер выбирает $B^{(m)}$, если

$$P\left(Z \mid B^{(m)}\right) = \max P\left(Z \mid B^{(m)}\right) \text{ по всем } B^{(m)}. \qquad (1.2)$$

При рассмотрении двоичной демодуляции предполагается передача только двух равновероятных сигналов $s_1(t)$ и $s_2(t)$. Следовательно, принятие двоичного решения на основе принципа максимального

правдоподобия, касающееся данного полученного сигнала, означает, что в качестве переданного сигнала выбирается $s_1(t)$, если

$$p(z \mid s_1) > p(z \mid s_2).$$

В противном случае считается, что передан был сигнал $s_2(t)$. Параметр z представляет собой величину $z(T)$, значение принятого сигнала до детектирования в конце каждого периода передачи символа $t = T$. Однако при использовании принципа максимального правдоподобия в задаче сверточного декодирования, в сверточном коде обнаруживается наличие памяти (полученная последовательность является суперпозицией текущих и предыдущих двоичных разрядов).

В 1967 году Витерби разработал и проанализировал алгоритм, в котором, по сути, реализуется декодирование, основанное на принципе максимального правдоподобия; однако в нем уменьшается вычислительная нагрузка за счет использования особенностей структуры конкретной решетки кода.

Алгоритм включает в себя вычисление меры подобия (или расстояния) между сигналом, полученным в момент времени t_i, и всеми путями решетки, входящими в каждое состояние в момент времени t_i. В алгоритме Витерби не рассматриваются те пути решетки, которые, согласно принципу максимального правдоподобия, заведомо не могут быть оптимальными. Если в одно и то же состояние входят два пути, выбирается тот, который имеет лучшую метрику; такой путь называется выживающим. Отбор выживающих путей выполняется для каждого состояния. Таким образом, декодер углубляется в решетку, принимая решения путем исключения менее вероятных путей.

Модифицированный Алгоритм декодирования Витерби. Каждый раз при возникновении ситуации, когда необходимо сделать выбор, по какой ветви решетки продолжать декодирования, декодер вместо исключения наименее вероятного пути будет просчитывать метрику для каждой ветви декодирования. Хотя сверточное кодирование подразумевает потоковые данные, в реальности каждое сообщение имеет конечную длину. Таким образом, решетка для входной последовательности для декодера будет конечной. И в конце концов получится массив оценок метрик для каждого возможного пути декодирования (за исключением тех, которые заведомо неправильные). Для экономии памяти, будем отбрасывать те ветви, которые имеют заданную длину (введем границу, определяющую максимальную длину ветви, для которой нужно просчитывать всевозможные пути декодирования) и наименьшую метрику из всех путей, имеющих такую же длину.

Несмотря на то, что модифицированный алгоритм декодирования Витерби ведет себя так же как и классический алгоритм декодирования Витерби для заданной границы, данная модификация поможет снизить вероятность ситуации, когда из-за одной неправильно определенной ветви декодирования, а следовательно, ошибочного исключения верной ветви, вся выходная последовательность будет неверной.

Оценки работы кодера и декодера. В зависимости от заданных порождающих многочленов происходит сложение по модулю 2 информационной последовательности и битов, находящихся в ячейках регистра сдвигов. Оценим быстродействие такого кодера. Для небольшого сообщения длиной около 3 бит кодер работает достаточно быстро. Однако, с ростом количества ячеек в регистре сдвигов растет и время работы кодера. На графиках 1 и 2 показаны зависимости времени работы кодера от увеличения длины информационной последовательности для регистров сдвига с количеством ячеек 3 и 4 соответственно.

Измерения проводились для длин информационной последовательности 3, 50, 150, 500, 750, 1000 и 1500 бит.

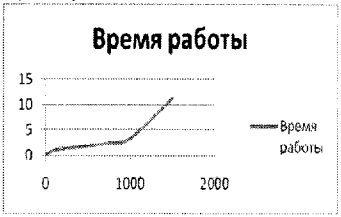

График 1. Зависимость времени работы кодера от длины информационной последовательности. Скорость кодирования $R = 1/2$

График 2. Зависимость времени работы кодера от длины информационной последовательности. Скорость кодирования $R = 1/3$

Для информационной последовательности длиной около 10000 бит время работы около 304мс и 316мс соответственно.

Оценка работы декодера Витерби и модифицированного декодера Витерби. Оценка работы декодера проводилась на информационной последовательности 11001. Информационная последовательность кодировалась с помощью порождающих многочленов. Далее в полу-

ченную кодовую последовательность вносились одиночные ошибки или пакеты ошибок (выделены жирным).

Рассмотрим кодер с порождающими многочленами: $g_1(X)$= {1000}; $g_2(X)$= {0110}; $g_3(X)$= {0111}. Занесем полученные данные в таблицу.

Скорость	Модифицированный алгоритм (ошибок в результате декодирования; время работы)	Классический алгоритм (ошибок в результате декодирования; время работы)
R = ½	Без ошибок	
	0;5	0;5
	1:1**0010111111**	
	0;7	1;6
	2:11010111**00**	
	1;5	1;5
	2:1101**10**1111	
	2;5	2-3;5-6
	4:1101**110100**	
	1;6	2-3;5-6
R = 1/3	Без ошибок	
	0;5	0;5
	1:**0**00111000010101	
	1;3	1;3
	1:10**1**111000010101	
	0;3	2;3
	2:**01**0111000010101	
	1;3	1;3
	2:10**1**011000010101	
	1;3	1;3
	3:**011**1110000101001	
	1;3	1;3
	3:10**100**11000010101	
	3;3	3;3
	4:**011**01000010101	
	2;3	2;3
	4:1**1100**1000010101	
	1;2	1;3
	5:**00**1**1**0**1**010101	
	2;3	2;3
	5:**01100**1000010101	
	2;3	3;3

Отбрасывание менее вероятных путей дает экономию в памяти при реализации декодирования. Так же такой метод декодирования помогает экономить время при декодировании больших входных последовательностей. Но, тем не менее, проанализировав данные из таблицы, можно сказать, что классический декодер Витерби удовлетворительно исправляет ошибки, начиная со скорости $R = 1/3$. Кроме того, место ошибки играет немаловажную роль. Если ошибка появилась в начале кодовой последовательности, то с большой вероятностью после работы декодера Витерби ответ мы получим неверный, содержащий один или более неверно декодированных символов.

Модифицированный декодер Витерби более устойчив к появлению ошибок. Как и следовало ожидать, на входных последовательностях небольшой длины (от 5 до 100 бит) модифицированный декодер Витерби работает примерно за то же время, что и классический алгоритм Витерби, но в некоторых ситуациях ошибок, которые допускает этот алгоритм при декодировании, меньше. На входных последовательностях большей длины модифицированному декодеру требуется больше времени, чем классическому декодеру. Но взамен большего времени, потраченного на декодирования, мы получаем более точный декодер, способный исправлять ошибки, которые классический алгоритм Витерби в большинстве случаев исправить не в состоянии.

В зависимости от реализации метода хранения значений метрик всевозможных путей, модифицированному алгоритму может не хватить памяти. Таким образом, мы вынуждены ограничивать длину входной последовательности. Следовательно, данную модификацию целесообразней использовать там, где необходимо точное декодирование сравнительно небольших входных последовательностей, например, обмен короткими текстовыми сообщениями или телефонная связь.

Классический же алгоритм декодирования почти не ограничен в длине входной последовательности, но в то же время обладает меньшей точностью, чем его модификация. Таким образом, классический декодер целесообразней использовать там, где важно быстро декодировать большие входные последовательности, например, в потоковом видео или аудио.

Библиографический список

1. Морелос-Сарагоса Р. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение / пер. с англ. В. Б. Афанасьева. – М.: Техносфера, 2006. – 320 с.

УДК 004(063)

Машины Тьюринга, сложность и случайность

А.Н. Гамова, А.С. Платонов
СГУ, г. Саратов

Машины Тьюринга бывают детерминированными и недетерминированными. Собственно вычисления происходят на детерминированных машинах Тьюринга, недетерминированные машины чаще используются для подтверждения разрешимости проблемы (распознавания языка). Модель вычислений на недетерминированной машине Тьюринга восходит к идеям Х. Эверетта о многозначности мира. Для недетерминированной машины Тьюринга это означает существование нескольких команд с одинаково левой частью и разными правыми частями. На каждой развилке дерева вычислений недетерминированной машины Тьюринга мир разбивается на несколько новых миров, в каждом из которых это событие заканчивается по-своему. Если произошло допускание на одной из веток дерева вычислений, работа недетерминированной машины заканчивается. Для того, чтобы машина не допустила входную строку, необходимо, чтобы это случилось на каждой ветке. Рассмотрим феномен проявления недетерминированности на примере задачи о рюкзаке.

Постановка задачи: В комнате находится N предметов разной массы и стоимости. Требуется заполнить ими рюкзак, максимизируя их суммарную стоимость, при условии, что общая масса предметов не больше заданного целого числа K.

Алгоритм:
итоговая стоимость = сумме стоимостей всех предметов в комнате;
ЦИКЛ
недетерминированный блок
масса ← 0, стоимость ←0;
ЦИКЛ выйти_или_не_выйти из цикла;
взять предмет из комнаты и поместить его в рюкзак;
масса ← масса + масса предмета;
стоимость ← стоимость + стоимость предмета;