Код Хаффмана и его построение без обхода дерева

5 января 2023 г. 1:21

Код Хаффмана - наилучший неравномерный побуквенный код

- P – матрица, в которой хранятся вероятности появления сообщений.
- L – матрица длин кодовых слов.
- C – матрица кодовых слов размером $M \times M$.
- T – матрица потомков узлов $M \times M$.

$$P = \begin{pmatrix} 0.5 \\ 0.25 \\ 0.125 \\ 0.125 \end{pmatrix}, C = \begin{pmatrix} - & - & - & - \\ - & - & - & - \\ - & - & - & - \\ - & - & - & - \end{pmatrix}, T = \begin{pmatrix} 0 & - & - & - \\ 1 & - & - & - \\ 2 & - & - & - \\ 3 & - & - & - \end{pmatrix}, L = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Нашли две минимальные вероятности в P (вершины 2 и 3), их нужно просуммировать и заменить одну из них на эту сумму, а вторую - на "прочерк".
Смотрим матрицу L, а именно строки 2 и 3 (две последние, т.к. нумерация с 0), соответствующие найденным минимальным вероятностям. Видим там нули. Значит, кодовые слова мы будем писать в матрицу C на 2 и 3 строку (при нумерации с 0) в столбец с индексом 0.
В матрицу T записываем номера потомков объединенного узла, т.е. все номера узлов, которые находились в строках 2 и 3.
Пересчитываем матрицу L - она содержит количество непустых значений для каждой строки матрицы C.

$$P = \begin{pmatrix} 0.5 \\ 0.25 \\ 0.25 \\ - \end{pmatrix}, C = \begin{pmatrix} - & - & - & - \\ - & - & - & - \\ 0 & - & - & - \\ 1 & - & - & - \end{pmatrix}, T = \begin{pmatrix} 0 & - & - & - \\ 1 & - & - & - \\ 2 & 3 & - & - \\ - & - & - & - \end{pmatrix}, L = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

Снова выбираем две минимальные вероятности (здесь - индексы 1 и 2). Заменяем на сумму и "прочерк".
Смотрим строки 1 и 2 в матрице T, видим там 1,2,3 => смотрим строки 1,2,3 в матрице L. В строке 1 написан 0 => пишем 0 в матрицу C в столбец 0 на строку 1 (это всё индексы). Далее видим в матрице L единицы, соответственно им дописываем единицы в матрицу C.
Обновляем матрицу T.
Пересчитываем матрицу L.

$$P = \begin{pmatrix} 0.5 \\ 0.5 \\ - \\ - \end{pmatrix}, C = \begin{pmatrix} - & - & - & - \\ 0 & - & - & - \\ 0 & 1 & - & - \\ 1 & 1 & - & - \end{pmatrix}, T = \begin{pmatrix} 0 & - & - & - \\ 1 & 2 & 3 & - \\ - & - & - & - \\ - & - & - & - \end{pmatrix}, L = \begin{pmatrix} 0 \\ 1 \\ 2 \\ 2 \end{pmatrix}$$

Суммируем оставшиеся две вероятности, заменяем их на 1 и "прочерк".
Обращаемся к 0 и 1 строкам матрицы T, видим 0,1,2,3 => смотрим строки 0,1,2,3 матрицы L. Видим 0 - пишем 0 в C на соответствующее место. Видим не 0 - дописываем единицы.
Обновляем матрицу T. На последней итерации в ней должна быть заполнена только строка с индексом 0.
Пересчитываем матрицу L.

$$P = \begin{pmatrix} 1.0 \\ - \\ - \\ - \end{pmatrix}, C = \begin{pmatrix} 0 & - & - & - \\ 0 & 1 & - & - \\ 0 & 1 & 1 & - \\ 1 & 1 & 1 & - \end{pmatrix}, T = \begin{pmatrix} 0 & 1 & 2 & 3 \\ - & - & - & - \\ - & - & - & - \\ - & - & - & - \end{pmatrix}, L = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 3 \end{pmatrix}$$

В итоге имеем кодовые слова согласно заполнению матрицы C.

Плюс такого алгоритма заключается в примитивности. Не надо работать с деревом, работаем только с четырьмя матрицами. Имплементация проще, чем в случае с обходом дерева. Проще написать на языке низкого уровня, на Ассемблере, например.

В каком случае код Хаффмана не избыточный? Т.е. когда он достигает энтропии?
Избыточность $R = \overline{l} - H = 0$
$\sum_i p_i l_i = -\sum_i p_i log\left(p_i\right)$
Ответ: в случае, если вероятности - степени двойки
Например, если вероятности 1/2, 1/4, 1/4, то избыточность = 0

Есть двоичный источник, т.е. алфавит {0,1}
Вероятность нуля 0,9
Вероятность единицы 0,1
(Максимальная энтропия была бы при вероятностях 0,5 и была бы равна 1)
Для этого случая будет код Хаффмана {0,1}
То есть по сути принимаем один бит и кодируем его так же одним битом
Это тот самый редкий случай, когда избыточность кода Хаффмана большая
Еще хуже будет, если вероятности = 0 и 1, тогда энтропия будет 0, и кодер будет вынужден тратить по 1 биту постоянно
Если сталкиваемся с подобной ситуацией, можем применить расширение алфавита
Например, от алфавита {0,1} переходим к алфавиту {1, 01, 001, 000}, считаем вероятности: {0,1; 0,9*0,1; 0,9*0,9*0,1; 0,9*0,9*0,9} и строим код Хаффмана для него