Проект по Структури от данни и програмиране

(спец. Компютърни науки, 2017/18 г.)

Алгоритъм на Хъфман за компресия на данни

Не е нова идеята да се предава информация по възможно най-икономичен начин. Например естествените говорими езици и писмени азбуки неизменно страдат от излишество. При тях обаче икономичното предаване на информация не е най-важната страна; макар и не оптимални от тази гледна точка те са удобни за използване от човек. За оптимално кодиране са разработени специални системи, каквито са например стенографската, морзовата азбука, азбуката за глухи, които са лишени от доста удобства. С навлизането на компютрите се появява възможност автоматично сравнително бързо да се "превежда" даден поток от информация на по-икономична азбука и обратно. Бързо намират приложение алгоритмите за компресиране на информация, а те от своя страна се доразработват и оптимизират, за да навлязат във всекидневна употреба. Всеки е използвал поне една универсална програма за компресиране (ARJ, ZIP, RAR, ACE, 7zip) и се е възползвал от компресии на мултимедия — звук (MP3, OGG, AAC), картина (GIF, JPEG, PNG), филмов клип (MPEG), дори и извън всекидневната работа с компютрите (компресия на звук по GSM). Алгоритмите за компресия имат стабилна математическа основа и стават все по-сложни и с по-добра степен на компресия с нуждата от тяхното прилагане.

Алгоритъм на Хъфман

Алгоритъмът на Хъфман, разгледан тук, е сравнително прост универсален алгоритъм за компресия без загуба на данни (за разлика от алгоритмите със загуба, стоящи в основата на MP3, например). При него се предполага, че е даден краен поток от числа в някакъв предварително фиксиран интервал. Ще считаме, че става дума за символи, кодирани със ASCII код, т.е. ще разглеждаме информацията като поредица от байтове (числа в интервала [0; 255]). Алгоритъмът се базира на простата идея, че най-често срещаните символи в поредицата трябва да се записват с най-малък брой битове. Така той построява нова азбука, която следва тази идея и след това превежда информацията в новата азбука. Кодирането е обратимо, т.е. по кодираната последователност може да се декомпресира — да се намери първоначалната поредица.

Построяване на дърво на Хъфман

Разглеждаме задачата за компресиране даден низ от символи. Искаме да построим двоично дърво, от което ще определим азбука за компресиране.
Алгоритъмът за построяване на дърво се състои от следните стъпки:

1. Създава се честотна таблица на низа — за всеки символ се записва броят на срещанията му.
2. Нека различните символи в низа са n на брой. Създаваме n дървета от по един възел, който съдържа наредена двойка: символ и число, означаващо броя на срещанията на символа в низа.

3. Намираме двете дървета с най-малки числа в корените. Създаваме ново дърво с двете намерени дървета като поддървета, а сумата от съответните им числа записваме в корена на новото дърво.
4. Повтаряме стъпка 3, докато не получим само едно дърво - дървото на Хъфман за дадения низ.

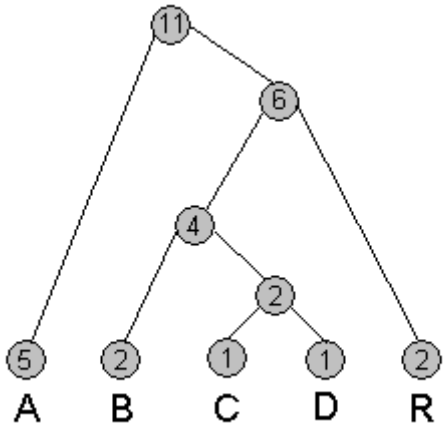
Така построено дървото е двоично и има точно n на брой листа, като на всяко листо отговаря един символ от честотната таблица. По начина на построение се вижда, че по-често срещаните символи се намират по-близо до корена от по-рядко срещаните. Това се вижда и в примера, даден по-долу.

Пример:

Нека имаме низа "ABRACADABRA". Честотната таблица за низа е:

Символ	Брой срещания
A	5
B	2
C	1
D	1
R	2

Строим дървото по следния начин:

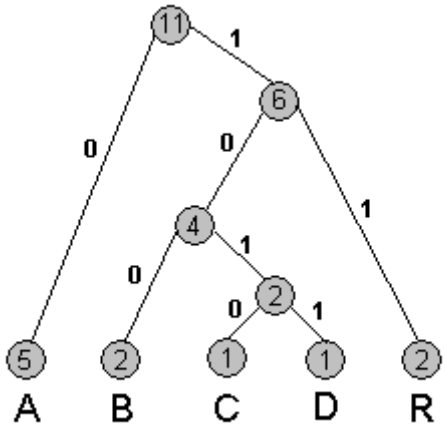


Построяване на азбуката по дървото на Хъфман

На всяко ребро от дървото съпоставяме двоична цифра 0 или 1: 0 за ляво ребро, 1 за дясно ребро. Така на всеки път от корена до някое листо отговаря двоичен низ. Тъй като всяко листо отговаря на символ от низа, можем да съпоставим на всеки символ двоичната последователност, която съответства на пътя от корена до листото на символа. Най-често срещаните символи са най-близко до корена, следователно на тях ще отговарят най-къси последователности. Обратно — на рядко срещаните символи съответстват дълги последователности.

Пример:

Продължаваме примера отгоре. След маркиране на ребрата на дървото с 0 и 1, то изглежда така:



Таблицата за кодиране е:

Символ	Код
A	0
B	100
C	1010
D	1011
R	11

След като получим таблицата за кодиране, извършваме кодиране на низа — всеки символ заместваме с неговия код. Така получаваме последователност от 0 и 1. Ако разбием тази последователност на блокове по 8 бита, можем да получим съответна последователност от байтове.

Пример:

```
ABRACADABRA → 0 100 11 0 1010 0 1011 0 100 11 0
            → 01001101010010110100110
            → 01001101 01001011 0100110
            → 77 75 38
```

От 11 символа (байта) = 8*11 бита = 88 бита получихме 23 бита компресирана информация - около 26% от оригиналния обем. Получихме четири пъти по-малко описание на "ABRACADABRA".

Декомпресиране на компресирана информация

Декомпресирането на данните става лесно при условие, че имаме дървото на Хъфман. Обхождаме едновременно двоичния низ и дървото, като всеки път като срещнем 0 завиваме наляво, а при 1 — надясно. Когато стигнем до листо, извеждаме съответния символ и рестартираме обхождането от корена на дървото. Така стъпка по стъпка получаваме първоначалния низ.

Пример

```
01001101010010110100110 → 0 100 11 0 1010 0 1011 0 100 11 0
                        → A B R A C A D A B R A
```

Задача

Да се напише програма, която въвежда съдържанието на текстов файл, построява честотна таблица и дървото на Хъфман и извежда в друг текстов файл двоичната последователност, която кодира оригиналния низ. Да се поддържат режим на работа,в който вместо извеждане на двоичната последователност, да се изведе поредица от числа от 0 до 255, получени от разбиване на блокове от по 8 бита. Да се пресметне степента на компресия (отношението на броя на битовете на компресираната и декомпресираната последователност — считаме, че всеки символ се кодира с един байт). По дървото на Хъфман да се възстанови оригиналният низ.

Създаденото дърво на Хъфман (**не** честотната таблица) да може да се извежда и въвежда в подходящ формат. Дървото на Хъфман да се пази в изходния текстов файл или в отделен файл.

Да се работи с произволен двоичен файл, като програмата реализираща кодирането на Хъфман трябва да може да приема като командни параметри следната информация:

-c[ompress] – параметър указващ че програмата ще работи в режим компресия;
-d[ecompress] – параметър указващ че програмата ще работи в режим декомпресия;
-i fname – параметър указващ името на входния файл (подлежащ на компресиране или декомпресиране);
-o fname – параметър указващ името на изходния файл;
В този случай е необходимо да се сравни, така реализираната архивираща програма със ZIP и RAR – при едни и същи входни файлове, какъв е резултата от работата на трите програми.

Бонус

Да се реализира възможност за ефективно кодиране на големи файлове или на потоци от данни, чрез имплементиране на Адаптивен алгоритъм на Хъфман. Програмата трябва да създава дървото на Хъфман без да е прочела всички данни (използвайки локалната честота на срещане на символите/байтовете) и да го модифицира в процеса на кодиране за да запази ефективността на алгоритъма.

Заключителни бележки

Описаният алгоритъм е един от най-простите алгоритми за компресиране. Това е универсален алгоритъм без загуба на информация за кодиране с променлива дължина. За декодиране е

необходимо да се пази допълнителна структура — в случая честотна таблица или дървото на Хъфман. За сравнение има алгоритми (LZ77, LZ78), които не се нуждаят от допълнителна структура, а строят такава динамично по време на компресия и декомпресия въз основа на самата информация. Направени са много подобрения на алгоритъма на Хъфман, подобряващи степента на компресиране. Последното е за сметка на усложняване на алгоритъма.