Лабораторна робота № 6.
Кодування послідовностей. Арифметичне кодування

Алгоритми Шеннона-Фано і Хаффмана в найкращому випадку не можуть кодувати кожний символ повідомлення менш ніж одним бітом інформації. Припустимо, що в повідомленні з 0 та 1 одиниці трапляються в 10 разів частіше. Ентропія такого повідомлення (що визначає верхню границю стиснення даних) H(X)≈0,469 (біт/сим) суттєво менше одиниці, тому кодування таких повідомлень оптимальними алгоритмами буде не достатньо ефективним. У таких випадках бажано використовувати алгоритми, що дозволяють кодувати символи повідомлення менш ніж 1 бітом інформації. Одним із найкращих таких алгоритмів є алгоритм арифметичного кодування.

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

На першому етапі слід обчислити або оцінити частоти виникнення кожного символу алфавіту. Найкращого результату можна досягти, прочитавши весь вхідний файл на першому проході алгоритму стиснення, що складається з двох проходів. Однак, якщо програма може отримати хороші оцінки частот символів з іншого джерела, перший прохід можна опустити.

Алгоритм кодування полягає в побудові інтервалу, що однозначно визначає конкретну послідовність значень д.в.в. Інтервали повідомлення будуються так. Якщо є відрізок повідомлення завдовжки n-1 символів, то для побудови відрізка повідомлення завдовжки n попередній інтервал розбивається на стільки частин, скільки можливих значень має д.в.в. Для знаходження початку і кінця нового інтервалу повідомлення до початку попереднього інтервалу необхідно додати значення добутків його ширини на відповідні границі відрізка поточного нового символу з таблиці символів і їхніх інтервалів (таблиці кодера). З отриманих інтервалів вибирається той, що відповідає конкретному повідомленню завдовжки n символів. Для побудованого таким чином інтервалу повідомлення знаходиться число, що належить цьому відрізку, як правило, це ціле число, розділене на мінімальний степінь 2. Це дійсне число і буде кодом даного повідомлення. Усі можливі коди - це числа, строго більші 0 і менші 1, тому 0 і десяткову крапку можна не враховувати.

У міру надходження символів повідомлення його інтервал звужується, відповідно кількість розрядів, необхідна для подання інтервалу збільшується. Більш імовірні символи меншою мірою звужують інтервал, ніж менш імовірні, і, отже, додають менше розрядів до результату.

Основна відмінність арифметичного кодування від алгоритмів Шеннона-Фано і Хаффмана полягає в його неперервності, тобто відсутності необхідності блокування повідомлення. Ефективність арифметичного кодування зростає із зростанням довжини повідомлення, проте й потребує значно більших обчислювальних ресурсів.

Приклад

Розглянемо три символи a_1, a_2 і a_3 з ймовірностями $P_1 = 0.4$, $P_2 = 0.5$ і $P_3 = 0.1$, відповідно. Інтервал [0,1) ділиться між цими трьома символами на частини пропорційно їх ймовірностям. Порядок слідування цих підінтервалів не суттєвий. Відповідні підінтервали - [0,0.4), [0.4,0.9) і [0.9,1.0). Щоб закодувати рядок «$a_2a_2a_2a_3$», починаємо з інтервалу [0,1). Перший символ a_2 скорочує цей інтервал, відкинувши від нього 40% на початку і 10% в кінці. Результатом буде інтервал [0.4,0.9). Другий символ a_2 скорочує інтервал [0.4,0.9) в тій же пропорції до інтервалу [0.6,0.85). Третій символ a_2 переводить його в [0.7,0.825). Нарешті, символ a_3 відкидає від нього 90% на початку, а кінцеву точку залишає без зміни. Виходить інтервал [0.8125,0.8250). Остаточним кодом нашого методу може служити будь-яке число з цього проміжку. (Зауважимо, що підінтервал [0.6,0.85) отримано з [0.4,0.9) за допомогою наступних перетворень його кінців: 0.4 + (0.9 - 0.4)*0.4 = 0.6 і 0.4 + (0.9 - 0.4)*0.9 = 0.85.

На цьому прикладі легко зрозуміти наступні кроки алгоритму арифметичного кодування:

1. Встановити «поточний інтервал» [0,1).
2. Повторити наступні дії для кожного символу ***s*** вхідного файлу.

2.1. Розділити поточний інтервал на частини пропорційно ймовірностям кожного символу.

2.2. Вибрати підінтервал, відповідний символу ***s***, і призначити його новим поточним інтервалом.

3. Коли весь вхідний файл буде оброблений, виходом алгоритму оголошується будь-яка точка, яка однозначно визначає поточний інтервал (тобто, будь-яка точка всередині цього інтервалу).

Декодування

Отриманий код дозволяє провести однозначне декодування. Алфавіт символів, ймовірності та інтервали символів відомі. Код інтерпретується як дійсне число виду $0.k_1k_2k_3...k_n$. Зчитуємо послідовно символи коду, напочатку отримуємо число $0.k_1$ — йому згідно таблиці відповідає інтервал, який його містить, а від так і встановюємо символ вхідної послідовності. Далі модифікуємо інтервал шляхом віднімання нижньої межі попереднього інтервалу та діленням на його довжину. Отримуємо нове дійсне число із діапазону [0,1). Знову шукаємо у таблиці відповідний символ і т. д.

Завдання

1.1. Для алфавіту із десяти символів, ймовірності яких задані згідно таблиці у завданні 1 для кожного варіанту, згенерувати послідовності символів довжиною 5 та 10 символів.
1.2. Для кожної із послідовностей побудувати арифметичний код.
1.3. Побудувати арифметичний код власного П.І.Б.
1.4. Декодувати коди.

Зміст лабораторної роботи

1. Для кожної послідовності (“5”,”10”,”ПІБ”) побудувати таблиці:
1.1 Вхідну таблицю для послідовності
Символ Ймовірність Область (нижня-верхня межа інтервалу)

1.2. Таблиця кодування
Символ Нижня межа інтервалу Верхня межа інтервалу

1.3. Таблиця декодуваня
Код-Нижня межа інтервалу Символ алфавіту Область

2. Порівняння сумарної довжини коду Хаффмана для послідовності символів **ПІБ** та довжини її арифметичного коду.

3. Висновки до роботи