МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИНСТИТУТ МЕТАЛЛУРГИИ, МАШИНОСТРОЕНИЯ И ТРАНСПОРТА
КАФЕДРА «МЕХАТРОНИКА И РОБОТОСТРОЕНИЕ (ПРИ ЦНИИ РТК)»

Дисциплина: Программирование на языках высокого уровня

Курсовая работа

Разработал:
ст. гр. 33335/2 Заморин Н.С.

Преподаватель Ананьевский М.С.

Санкт-Петербург

2018

Оглавление

Формулировка задачи

Дан массив чисел *a[0 ... n]*. Требуется найти такой его подотрезок *a[r ... m]*, что сумма элементов, входящих в него, максимальна.

$$\max_{0 \le r \le m \le n} \sum_{i=r}^{m} a[i]$$

Отметим, что в случае, когда все числа положительные, искомый подотрезок совпадает с данным массивом, а, когда все числа отрицательные, искомый подотрезок содержит лишь один элемент - максимальный элемент массива.

Описание алгоритма Кадана

Алгоритм предложенный Джеем Каданом в 1984 г. выглядит следующим образом. Будем идти по массиву и накапливать текущую сумму элементов в переменной *s_cur*. Если сумма элементов станет отрицательной, то присвоим ей значение 0, и вновь начнем накапливать текущую сумму. Утверждается, что максимальное значение переменной *s_cur* и будет суммой элементов искомого отрезка.

Докажем этот алгоритм.

Рассмотрим момент времени, когда сумма *s_cur* стала отрицательной. Это означает, что, стартовав с нулевой частичной суммы, мы в итоге пришли к отрицательной частичной сумме — значит, и весь этот префикс (или суффикс) массива имеет отрицательную сумму. Следовательно, от всего этого префикса (или суффикса) массива в дальнейшем не может быть никакой пользы: он может дать только отрицательную прибавку к ответу.

Однако этого недостаточно для доказательства. В алгоритме мы, фактически, ограничиваемся в поиске ответа только теми отрезками, которые начинаются непосредственно после мест, когда случалось *s_cur <0*.

Рассмотрим произвольный отрезок *[r ... m]*, где *r>p+1* (*p* – последняя «критическая» позиция, в которой *s_cur <0*). Так как отрезок *[p+1 ... r-1]* не

содержит в себе критических позиций, то сумма элементов этого отрезка больше или равна нулю. Соответственно, сумма элементов отрезка *[p+1 ... m]* будет больше или равна сумме элементов отрезка *[r ... m]* ($\sum_{i=p+1}^{m} a[i] - \sum_{i=r}^{m} a[i] = \sum_{i=p+1}^{r-1} a[i] \geq 0$, откуда следует $\sum_{i=p+1}^{m} a[i] \geq \sum_{i=r}^{m} a[i]$). Таким образом, действительно можно ограничиться в поиске ответа только теми отрезками, которые начинаются непосредственно после мест, когда случалось *s_cur <0*.

Реализация алгоритма

Псевдокод

```
s_current = 0
s_max = 0
for i = 0 to n-1
  s_current = max(s_current + x[i], 0)
  s_max = max(s_current, s_max)
```

На языке C

```
unsigned int left_current = 0;
unsigned int right_answer = 0;
unsigned int left_answer = 0;
int s_current = 0;
int s_answer = array[0];
for (unsigned int i = 0; i < size; i++){
    s_current += array[i];
    if(s_current > s_answer){
        left_answer = left_current;
        right_answer = i;
        s_answer = s_current;
    }
    if(s_current < 0){
        s_current = 0;
        left_current = i + 1;
    }
}
```

Анализ алгоритма

Сложность алгоритма

Операторы внутри цикла выполняются n раз, где n – размер входных данных, то есть размер массива. Соответственно, сложность алгоритма составляет $O(n)$.