A1

Задание А1

Даны два рекурсивных алгоритма обработки целочисленного массива A размера n:

```
algorithm1(A, n)
1  if n ≤ 20
2      return A[n]
3  x = algorithm1(A, n - 5)
4  for i = 1 to ⌊n/2⌋
5      for j = 1 to ⌊n/2⌋
6          A[i] = A[i] - A[j]
7  x = x + algorithm1(A, n - 8)
8  return x
```

```
algorithm2(A, n)
  if n ≤ 50
      return A[n]
  x = algorithm2(A, ⌊n/4⌋)
  for i = 1 to ⌊n/3⌋
      A[i] = A[n - i] - A[i]
  x = x + algorithm2(A, ⌊n/4⌋)
  return x
```

1. **(2 балла)** Для каждого из представленных алгоритмов составить *рекуррентное соотношение*, которое выражает их временную сложность $T(n)$. Обратите внимание, что рекуррентное соотношение должно давать полное представление о сложности алгоритма, т.е., охватывать как рекурсивную, так и нерекурсивную ветку вычислений. Предполагается, что все арифметические операции выполняются за постоянное время.
2. **(5 баллов)** Вычислите *асимптотическую точную границу* **θ(f(n))** временной сложности для каждого из представленных алгоритмов, если это возможно. В случае невозможности формирования асимптотической точной границы, представить отдельно верхнюю и нижнюю границы. Обоснуйте свой ответ с помощью метода подстановки, дерева рекурсии, или индукции.

Для первого алгоритма

$T(n) = T(n-5) + T(n-8) + O(n^2)$

$T(n) \le O(2^n)$

$T(n) \le C2^n$

$T(n-5) \le C2^{n-5}$

$T(n-8) \le C2^{n-8}$

$$\frac{C2^n}{32} + \frac{C\cdot 2^n}{256} + C_1 n^2 \le C2^n$$

$$C2^n\left(\frac{8}{256} + \frac{1}{256}\right) + C_1 n^2 \le C2^n$$

$$C\cdot 2^n \cdot \frac{9}{256} + C_1 n^2 \le C\cdot 2^n$$

Рассуждение I: 2^n растёт быстрее чем $n^2 \Rightarrow C_1 > 0$ и мы по сути можем "избавиться" от этого коэф. и обратить внимание на C.

$\frac{C\cdot 2^n \cdot 9}{256} \le C\cdot 2^n$ - такое возможно

$T(n) \ge \Omega(2^n)$

$T(n) \ge C\cdot 2^n$

— " —

$$\frac{C\cdot 2^n \cdot 9}{256} + C_1 n \ge C\cdot 2^n$$

Воспользуемся рассуждением I

$\frac{C\cdot 2^n 9}{256} \ge C\cdot 2^n$ - дробь

не может быть больше "1" $\Rightarrow$ такого быть не может, нижняя граница отсутствует $\Rightarrow$ точной границы нет.

Ответ: $O(2^n)$, $\Omega(n)$

$T(n) \ge \Omega(n)$

$T(n) \ge Cn$

$T(n-5) \ge C(n-5)$

$T(n-8) \ge C(n-8)$

$T(n) \ge C(n-5) + C(n-8) + C_1 n^2 \ge Cn$

$2Cn - 13C + C_1 n^2 \ge Cn$

$C(2n - 13) + C_1 n^2 \ge Cn$

$2Cn + C_1 n^2 \ge Cn$

$2C + C_1 n \ge C$

$C + C_1 n \ge 0$

$C_1 n \ge -C$

$C_1 > 0;\ C > 0$

A1

19 октября 2023 г. 15:22

Даны два рекурсивных алгоритма обработки целочисленного массива A размера n:

```
algorithm1(A, n)
1  if n ≤ 20
2      return A[n]
3  x = algorithm1(A, n - 5)
4  for i = 1 to ⌊n/2⌋
5      for j = 1 to ⌊n/2⌋
6          A[i] = A[i] - A[j]
7  x = x + algorithm1(A, n - 8)
8  return x
```

```
algorithm2(A, n)
  if n ≤ 50
      return A[n]
  x = algorithm2(A, ⌊n/4⌋)
  for i = 1 to ⌊n/3⌋
      A[i] = A[n - i] - A[i]
  x = x + algorithm2(A, ⌊n/4⌋)
  return x
```

1. **(2 балла)** Для каждого из представленных алгоритмов составить *рекуррентное соотношение*, которое выражает их временную сложность $T(n)$. Обратите внимание, что рекуррентное соотношение должно давать полное представление о сложности алгоритма, т.е., охватывать как рекурсивную, так и нерекурсивную ветку вычислений. Предполагается, что все арифметические операции выполняются за постоянное время.
2. **(5 баллов)** Вычислите *асимптотическую точную границу* $\Theta(f(n))$ временной сложности для каждого из представленных алгоритмов, если это возможно. В случае невозможности формирования асимптотической точной границы, представить отдельно верхнюю и нижнюю границы. Обоснуйте свой ответ с помощью метода подстановки, дерева рекурсии, или индукции.

Для второго алгоритма: $T(n) = 2T(n/4) + O(n)$

$T(n) \le cn$

$T(n/4) \le \frac{cn}{4}$

$T(n) \le 2 \cdot \frac{cn}{4} + c_1 n \le cn$

$\frac{cn}{2} + c_1 n \le cn$

$\frac{c}{2} + c_1 \le c$

$c_1 \le \frac{c}{2}$ - верно

$T(n) \ge \Omega(n)$

$T(n) \ge cn$

$T(n/4) \ge \frac{cn}{4}$

$T(n) \le 2 \cdot \frac{cn}{4} + c_1 n \ge cn$

$\frac{c}{2} + c_1 \ge c$

$c_1 \ge \frac{c}{2}$ - верно

Ответ: точная граница возможна $\Theta(n)$

A2

19 октября 2023 г. 15:22

Дан ряд рекуррентных соотношений, которые описывают временную сложность некоторых рекурсивных алгоритмов (при $n=1$ во всех случаях принимаем $T(1) = 1$):

- $T(n) = 7 \cdot T(n/3) + n^2$,
- $T(n) = 4 \cdot T(n/2) + \log n$,
- $T(n) = 0{,}5 \cdot T(n/2) + 1/n$,
- $T(n) = 3 \cdot T(n/3) + n/2$.

1. **(2 балла)** Для приведенных рекуррентных соотношений вычислите асимптотическую точную границу временной сложности, применяя основную теорему о рекуррентных соотношениях (*master-теорему*), если это возможно. Если применение master-теоремы невозможно, поясните почему.
2. **(3 балла)** Для рекуррентного(-ых) соотношения(-ий), не разрешимых с помощью master-теоремы, определите *возможную* асимптотическую верхнюю границу, используя метод подстановки.

1) $T(n) = 7 \cdot T(n/3) + n^2 = 7 \cdot T(n/3) + O(n^2)$

$c = 2 \quad \log_3 7 \approx 1, \ldots$

$c > \log_3 7 = O(n^2) = \Theta(n^2)$

2) Не возможно подсчитать с помощью master-теоремы

хотя, если заменить $\log n$ на $\sqrt{n}$, т.к. $\sqrt{n}$ растет быстрее, то

$T(n) = 4T(n/2) + \sqrt{n} = 4T(n/2) + O(n^{\frac{1}{2}})$

$c = \frac{1}{2} \quad \log_2 4 = 2$

$\frac{1}{2} < 2 \Rightarrow O(n^{\log_2 4}) = O(n^2) = \Theta(n^2)$

3) $T(n) = 0{,}5\,T(n/2) + 1/n = 0{,}5\,T(n/2) + O(n^{-1})$

не возможно, т.к. $c < 0$

4) $T(n) = 3T(n/3) + n/2 = 3T(n/3) + O(n)$

$c = 1 \quad ; \log_3 3 = 1$

$1 = 1 \Rightarrow O(n \log n) = \Theta(n \log n)$

- $T(n) = 4 \cdot T(n/2) + \log n$,
- $T(n) = 0{,}5 \cdot T(n/2) + 1/n$,

$T(n) = 4 \cdot T(n/2) + \log n$

$T(n) \le O(n^3)$

$T(n) \le cn^3$

$T(n/2) \le \frac{cn^3}{8}$

$T(n) \le \frac{4 \cdot cn^3}{8} + \log n \le cn^3$

$\frac{cn^3}{2} + \log n \le cn^3 \Rightarrow n^3$ растет быстрее чем $\log n \Rightarrow$

Ответ: $O(n^3)$

$O(n^3)$

° $\frac{cn^3}{2}$ + вода $\leq cn^3$ $\Rightarrow n^3$ расчет быстрее чем вода $\Rightarrow$ "
$\Rightarrow \frac{cn^3}{2} \leq cn^3$ - возможно

A2

19 октября 2023 г. 15:22

- $T(n) = 4 \cdot T(n/2) + \log n,$
- $T(n) = 0{,}5 \cdot T(n/2) + 1/n,$

$T(n) = 0{,}5\,T(n/2) + \frac{1}{n}$

$T(n) \le O(n)$

$T(n) \le cn$

$T(n/2) \le \frac{cn}{2}$

$T(n) \le \frac{5}{10} \cdot \frac{cn}{2} + \frac{1}{n} \le cn$

$\frac{cn}{4} + \frac{1}{n} \le cn$

n растёт быстрее чем $\frac{1}{n}$

$\frac{cn}{4} \le cn$ – возможно

Ответ: $O(n)$

А3

19 октября 2023 г. 15:22

Вы планируете разработать алгоритм ***MULT***, предназначенный для умножения двух квадратных матриц ***A*** и ***B*** размера ***N × N*** и асимптотически более эффективный, чем алгоритм Штрассена. Разрабатываемый алгоритм будет также использовать стратегию «Разделяй-и-властвуй».

1. Исходные матрицы ***A*** и ***B*** разделяются на фрагменты размера ***N/4 × N/4*** для дальнейшей рекурсивной обработки.
2. Временные затраты на выполнение шагов **DIVIDE** и **COMBINE** вместе составляют $\theta(N^2)$.

Таким образом, временная сложность алгоритма будет описываться следующим рекуррентным соотношением: $T(N) = a \cdot T(N/4) + \theta(N^2)$, где коэффициент ***a*** отвечает за количество решаемых подзадач. Например, для алгоритма Штрассена в соответствии с рекуррентным соотношением $T(N) = 7 \cdot T(N/2) + \theta(N^2)$ известно, что для каждой задачи решается *7* подзадач вдвое меньшего размера.

*В каком диапазоне должен находиться параметр **a** разрабатываемого вами алгоритма **MULT**, для того, чтобы он был асимптотически более эффективным по времени в сравнении с алгоритмом Штрассена? Обоснуйте свой ответ.*

$T(N) = a \cdot T(N/4) + \Theta(N^2)$

$T(N) = 7 \cdot T(N/2) + \Theta(N^2)$

Мы знаем, что сложность алгоритма Штрассена $O(n^{\log_2 7})$ – было доказано на лекции

Асимптотически более эффективным это тот, кто растет быстрее. В нашем случае это тогда, что $< n^{\log_2 7}$

По мастер теореме $c : 2$; $\log_4 a$

Скажем, что $\log_2 7 = 2{,}8074 \Rightarrow n^{2{,}8074}$

Наша функция должна быть меньше, чем $n^{2{,}8074}$

$n^x < n^{2{,}8074}$

$x \in (c : 2{,}8074)$

1) $c > \log_4 a$, то $O(n^2)$
2) $c = \log_4 a$, то $O(n^2 \log n)$
3) $c < \log_4 a$, то $O(n^{\log_4 a})$

Итак, нам подходит 1) случай, т.к. $2 < 2{,}8074$. $\Rightarrow$

$\Rightarrow a \in (1; 16)$

2) случай нам тоже подходит, т.к. $O(n^2 \log n)$ асимптотически более эффективен чем $O(n^{\log_2 7}) \Rightarrow a = 16$

3) случай нам подходит тогда, когда $O(n^{\frac{1}{2}\log_2 a}) < O(n^{\log_2 7})$

$\frac{1}{2}\log_2 a < \log_2 7$

$\log_2 a < 2 \log_2 7$

$a < 49$

Ответ: асимптотически более эффективен $T(n) = aT(N/4) + \theta(n^2)$ при $a \in (1; 49)$

A4

19 октября 2023 г. 15:22

22 октября 2023 г. 21:58

Длина интервала - кол-во эл. в соответствующем множестве.

n-интервалов,

выдаёт длину самого длинного перекрытия между парой интервалов и интерв.

Асимп. сложность алгоритма - O(nlogn), n-кол-во интервалов

В структуре:

длина, поиск перекрытия

Использование встроенных методов сортировки, поиска и проч. недопускается.

Вывод:

a - длина макс. перекрытия

x y - гран. интервала.

Если интервалов несколько, то выводится тот, у которого наименьшие границы.

Merge Sort

Разделить пол из n эл

на две по n/2 эл. в каждой