Домашняя работа 1

Пасечник Даша

на 15.02.2019

Задача 1

Построим двухленточную машину Тьюринга - T. Сначала T проходит по слову, написанному на первой ленте, и переписывает его на вторую. Затем головка первой ленты возращается к началу слова. Далее головка первой ленты идет вправо, а головка второй ленты влево, и на каждом шаге переход осуществляется только если обе головки указывают на одинаковые символы. Таким образом, если слово является палиндромом, то T проидет по слову сначала до конца, иначе - остановится внутри слова.
Приведем формальное описание T:

$T = (A, Q, Q_f, q_0, \delta, \Lambda)$
$Q = \{q_0, q_1, q_2, q_3\}$
$A = \{\lambda, a, b\}$
$Q_f = \{q_3\}$

$\delta:$

$$\delta\,(q_0, a, \Lambda) = (q_0, a, a, +1, +1)$$
$$\delta\,(q_0, b, \Lambda) = (q_0, b, b, +1, +1)$$
$$\delta\,(q_0, \Lambda, \Lambda) = (q_1, \Lambda, \Lambda, -1, 0)$$
$$\delta\,(q_1, a, \Lambda) = (q_1, a, \Lambda, -1, 0)$$
$$\delta\,(q_1, b, \Lambda) = (q_1, b, \Lambda, -1, 0)$$
$$\delta\,(q_1, \Lambda, \Lambda) = (q_2, \Lambda, \Lambda, +1, -1)$$
$$\delta\,(q_2, a, a) = (q_2, a, a, +1, -1)$$
$$\delta\,(q_2, b, b) = (q_2, b, b, +1, -1)$$
$$\delta\,(q_2, \Lambda, \Lambda) = (q_3, \Lambda, \Lambda, 0, 0)$$

Задача 2

Доказать, что следующие определения перечислимого множества $X \subset N$ эквивалентны:

- Существует алгоритм, печатающий все элементы множества (в любом порядке и со сколь угодно большими паузами между элементами).
- Множество является областью определения некоторой вычислимой функции.
- Множество является областью значений некоторой вычислимой функции.

($1 \Rightarrow 3$) Если существует алгоритм, печатающий все элементы множества, то множество является областью значений некоторой вычислимой функции.
Построим функцию $f(n)$, область значений которой - X.
На входе n запустим алгоритм, печатающий элементы X, и будем считать число напечатанных элементов. Когда будет напечатан n-ый элемент - выдадим его в качестве значения функции f на входе n. Множество значений функции f - X.

($3 \Rightarrow 2$) Если множество является областью значений некоторой вычислимой функции, то множество является областью определения некоторой вычислимой функции.
Пусть функция $f(n)$ имеет область значений X. Построим функцию $g(x)$ такую, что её область определения - X. На входе x будем перебирать все натуральные числа и подавать на вход f(n). Если $f(n) = x$, то $g(x) = 1$, иначе - $g(x)$ не определена в x.
Получим, что X - область определения g.

($2 \Rightarrow 3$) Если множество является областью значений некоторой вычислимой функции, то множество является областью определения некоторой вычислимой функции.
Пусть g(x) - функция, область значений которой - X. Переопределим g(x) так, что для каждого x из X $g(x) = x$. Теперь X - область значений некоторой вычислимой функции.

($3 \Rightarrow 1$) Если множество является областью определения некоторой вычислимой функции, то существует алгоритм, печатающий все элементы множества.
Пусть функция $f(n)$ имеет область значений X. Для каждого натурального числа, если функция f(n) определена, то печатаем результат. Это и будет алгоритм, печатающий все элементы множества X и только их.

Эквивалентность доказана.

Задача 3

Дан массив из n элементов, на которых определено отношение равенства (например, речь может идти о массиве картинок или музыкальных записей). Постройте алгоритм, который в «потоковом режиме обработки данных» определяет, есть ли в массиве элемент, повторяющийся больше $n/2$ раз. Считается, что в вашем распоряжении есть память объемом $O(logn)$ битов.

Введем две дополнительных переменных ans и $counter$: в переменной ans в каждый момент времени находится элемент массива предположительно встречающийся больше $n/2$ раз, $counter$ - это счетчик.

Алгоритм:

1. В ans кладем первый элемент массива, $counter = 0$.

2. При первом проходе по массиву на каждом шаге выполняем следующие действия:

- Если $counter = 0$, записываем текущий элемент массива в ans, $counter = 1$.

- Если $counter! = 0$, сравниваем ans с текущим элементом массива: если совпадают, то $counter+ = 1$, иначе $counter- = 1$.

3. Если искомый элемент существует, то после прохода по массиву он будет лежать в ans. Поэтому обнуляем $counter$ и делаем второй проход по массиву: сравниваем каждый элемент с ans. Если текущий элемент совпадает с ans, $counter+ = 1$.

4. Если $counter > n/2$, выводим ans.

Сложность данного алгоритма $O(n)$, а требуемая дополнительная память — $O(1)$.

Задача 4

На вход подается описания n событий в формате (s,f) — время начала и время окончания. Требуется составить расписание для человека,который хочет принять участие в максимальном количестве событий.

Алгоритм 1:

Выберем событие кратчайшей длительности, добавим его в расписание, исключим из рассмотрения события, пересекающиеся с выбранным. Продолжим делать то же самое далее.

Контрпример: Пусть есть события (1, 5), (6, 10), (4, 7). Алгоритм 1 выберет событие (4, 7), как кратчайшее и исключит из рассмотрения (1, 5) и (6, 10), как пересекающиеся с ним. Больше событий не осталось, следовательно, результат работы алгоритма: 1. Очевидно, что существует расписание, при котором можно посетить 2 события: (1, 5) и (6, 10), т.к. они не пересекаются. Вывод: алгоритм не оптимальный.

Алгоритм 2:

Выберем событие, наступающее раньше всех, добавим его в расписание, исключим из рассмотрения события, пересекающиеся с выбранным. Продолжим делать то же самое далее.

Контрпример: Пусть есть события (1, 10), (2, 3), (4, 5). Алгоритм 2 выберет событие (1, 10), как наступающее раньше всех и исключит из рассмотрения (2, 3) и (4, 5), как пересекающиеся с ним. Больше событий не осталось, следовательно, результат работы алгоритма: 1. Очевидно, что существует расписание, при котором можно посетить 2 события: (2, 3) и (4, 5), т.к. они не пересекаются. Вывод: алгоритм не оптимальный.

Алгоритм 3:

Выберем событие, завершающееся раньше всех, добавим его в расписание, исключим из рассмотрения события, пересекающиеся с выбранным. Продолжим делать то же самое далее.

Докажем от противного, что Алгоритм 3 работает корректно:

Среди всех примеров, где алгоритм работает неоптимально, выберем тот, который содержит минимальное число событий. Посмотрим на первое событие, выбранное алгоритмом: это событие, которое раньше всех заканчивается (Обозначим его C1). Пусть оно не входит в оптимальное решение. Значит, первое событие в оптимальном решении (обозначим его C2) заканчивается позже (не раньше) события C1. Тогда мы можем поменять C2 на C1 в оптимальном решении, т.к. это не изменит число событий в оптимальном решении и не приведет к пересечениию между событиями. Теперь решение, составленное алгоритмом 3, и оптимальное начинаются с одного и того же события. Отбросим его. Получим пример, в котором решение, составленное Алгоритмом 3 все еще не оптимальное, а рассматриваемых событий меньше, чем в предыдущем. Получили противоречие с тем, что изначально рассматриваемый пример был минимальным.

Т.о. Алгоритм 3 всегда работает корректно, так что выбираем его. Предварительная сортировка по времени конца события $O(n \log n)$, проход по массиву: $O(n)$. Следовательно, сложность алгоритма: $O(n \log n)$.

Задача 5

Найдите явное аналитическое выражение для производящей функции чисел BR_{4n+2} правильных скобочных последовательностей длины 4n + 2.

Обозначим BR_{4n+2} через T_{2n+1}. Для чисел Каталана известно реккурентное соотношение:

$$T_n = T_0 T_{n-1} + T_1 T_{n-2} + ... + T_{n-1} T_0$$

$$T_0 = 1$$

Производящая функция для чисел Каталана длины $4n + 2$, т.е. содержащих $2n + 1$ открывающуюся скобку имеет вид:

$$A(x) = T_1 + T_3 x + T_5 x^2 + ... + T_{2n+1} x^n + ...$$

Рассмотрим также производящую функцию для чисел Каталана длины $4n$:

$$B(x) = T_0 + T_2x + T_4x^2 + ... + T_{2n}x^n + ...$$

Распишем их произведение:

$$AB = (T_1 + T_3x + T_5x^2 + ...)(T_0 + T_2x + T_4x^2 + ...)$$

$$AB = T_1T_0 + (T_3T_0 + T_1T_2)x + (T_1T_4 + T_2T_3 + T_5T_0)x^2 + ...$$

Заметим, что:

$$T_2 = T_1T_0 + T_0T_1 => T_1T_0 = T_2/2$$

Аналогично:

$$T_3T_0 + T_1T_2 = T_4/2$$

$$T_1T_4 + T_2T_3 + T_5T_0 = T_6/2$$

$$...$$

Тогда:

$$AB = \frac{T_2}{2} + \frac{T_4}{2}x + \frac{T_6}{2}x^2 + ... + \frac{T_{2n}}{2}x^{n-1+...}$$

Домножим обе части равества на 2x, тогда справа окажется B(x) без первого члена T_0:

$$2xAB = T_2x + T_4x + ... + T_{2n}x^n + ...$$

$$2xAB = B - T_0$$

Т.к. $T_0 = 1$:

$$2xAB = B - 1 \tag{1}$$

Теперь распишем $B^2(x)$:

$$B^2 = (T_0 + T_2x + T_4x^2 + ...)(T_0 + T_2x + T_4x^2 + ...)$$

$$B^2 = T_0^2 + (T_0T_2 + T_2T_0)x + (T_0T_4 + T_2T_2 + T_4T_0)x^2 + ...$$

Заметим, что:

$$T_0^2 = T_1$$

$$T_0T_2 + T_2T_0 = T_3 - T_1^2$$

$$T_0T_4 + T_2T_2 + T_4T_0 = T_5 - (T_1T_3 + T_3T_1)$$

$$...$$

Тогда:

$$B^2 = T_1 + T_3x - T_1^2x + T_5x^2 - (T_1T_3 + T_3T_1)x^2 + ...$$

Распишем A^2:

$$A^2 = (T_1 + T_3x + T_5x^2 + ...)(T_1 + T_3x + T_5x^2 + ...)$$

$$A^2 = T_1^2 + (T_1T_3 + T_3T_1)x + (T_1T_5 + T_3T_3 + T_5T_1)x^2 + ...$$

Тогда выражение для B^2 представимо в виде:

$$B^2 = A - A^2x \tag{2}$$

Объединим (1) и (2) в систему. Напомним, что A - производящая функция для чисел Каталана длины $4n+2$, B - производящая функция для чисел Каталана длины $4n$. Нужно найти А, но если искать напрямую получим уравнение 4 степени. Так что вместо этого найдем сначала В, затем А из (2).

$$B^2 = A - A^2x$$

$$B - 2xAB = 1 \Rightarrow A = \frac{B-1}{B \cdot 2x}$$

$$B^2 = \frac{B-1}{2xB} - \frac{(B-1)^2}{4B^2x^2}x$$

$$4xB^4 = (B-1)(2B - B + 1)$$

$$4xB^4 = (B-1)(B+1)$$

$$4xB^4 = B^2 - 1$$

Решаем квадратное уравнение:

$$B_{12}^2 = \frac{1 \pm \sqrt{1 - 16 \cdot x}}{8x}$$

Какое из решений выбрать?? Заметим, что $B(x = 0) = T_0 = 1$. Тогда:

$$8xB^2 = 1 \pm \sqrt{1 - 16x}$$

$$0 = 1 \pm \sqrt{1}$$

Чтобы сохранялось равенство нужен знак -:

$$B^2 = \frac{1 - \sqrt{1 - 16x}}{8x}$$

Решим (2) как квадратное уравнение отностительно А:

$$A = \frac{1 - \sqrt{1 - 4xB^2}}{2x}$$

Знак - взят по тем же причинам.

Теперь подставляем выражение для B^2 и получаем ответ:

$$A = \frac{1 - \sqrt{1 - \frac{1 - \sqrt{1-16x}}{2}}}{2x}$$

Задача 6

Оцените трудоемкость рекурсивного алгоритма, разбивающего исходную задачу размера n на три задачи размером $\lceil \frac{n}{\sqrt{3}} \rceil - 5$, используя для этого $10\frac{n^3}{\log n}$ операций.

Строим реккуренту:

$$T(n) = 3T\left(\left[\frac{n}{\sqrt{3}}\right] - 5\right) + \Theta\left(10\frac{n^3}{\log n}\right)$$

Асимптотически нам неважны константы и целое округление:

$$T(n) = 3T\left(\frac{n}{\sqrt{3}}\right) + \Theta\left(\frac{n^3}{\log n}\right)$$

Найдем зависимость $\Theta(...)$ от глубины рекурсии:

$$\frac{n^3}{\log n} \Rightarrow 3 \cdot \frac{\frac{n^3}{(\sqrt{3})^3}}{\log\left(\frac{n}{\sqrt{3}}\right)} = \frac{\frac{n^3}{\sqrt{3}}}{\log\left(\frac{n}{\sqrt{3}}\right)} \Rightarrow 9 \cdot \frac{\frac{n^3}{(\sqrt{3})^3}}{\log\left(\frac{n}{(\sqrt{3})^2}\right)} = \frac{\frac{n^3}{\sqrt{3}}}{\log\left(\frac{n}{(\sqrt{3})^2}\right)}$$

Заметим, что k-ый элемент имеет вид:

$$\frac{\frac{n^3}{(\sqrt{3})^k}}{\log\left(\frac{n}{(\sqrt{3})^k}\right)}$$

Глубина рекурсии $\log n$, т.к. $T(n)$ выражается через $T(n/const)$. Поэтому трудоемкость алгоритма выражается суммой:

$$T(n) = \sum_{k=0}^{\log(n)} \frac{\frac{n^3}{(\sqrt{3})^k}}{\log\left(\frac{n}{(\sqrt{3})^k}\right)}$$

Т.к. все слагаемые положительны, оценим её снизу первым членом:

$$T(n) = \Omega(\frac{n^3}{\log n})$$

Докажем, что сверху выполняется та же оценка по индукции по глубине рекурсии.

Б.И.: $k = 0$, тогда сумма представляет собой первое слагаемое и равна $\frac{n^3}{\log n}$. Доказано.

Ш.И.: Пусть доказано для $k < m$, докажем для $k = m$:

По предположению индукции верно:

$$\exists C_1 > 0 : \exists N_1 > 0 : \forall n \geq N_1 \mapsto \sum_{k=0}^{m-1} \frac{\frac{n^3}{(\sqrt{3})^k}}{\log\left(\frac{n}{(\sqrt{3})^k}\right)} \leq C_1 \frac{n^3}{\log n}$$

Преобразуем слагаемое с номером m:

$$\frac{\frac{1}{(\sqrt{3})^m}}{\log\left(\frac{n}{(\sqrt{3})^m}\right)} = \frac{1}{(\sqrt{3})^m(\log n - m\cdot\log(\sqrt{3}))}$$

Теперь оценим его:

$$\frac{1}{(\sqrt{3})^m(\log n - m\cdot\log(\sqrt{3}))} \leq C\frac{1}{\log n}$$

$$\log n \leq C(\sqrt{3})^m(\log n - m\log\sqrt{3})$$

$$m\log\sqrt{3} \leq \Big(C\cdot(\sqrt{3})^m - 1\Big)\log n$$

$$m \leq \frac{\big(C(\sqrt{3})^m - 1\big)}{\log\sqrt{3}}\log n$$

Для сколь угодно большого m найдутся достаточно большие C и N такие, что неравенство выполняется. Тогда слагаемое с номером m можно оценить сверху $C\frac{1}{\log n}$. Сложим это неравенство с неравенством из предположения индукции: для $_2 > C_1 + C$ и $N_2 = max(N_1, N)$ выполняется:

$$\exists C_2 > 0 : \exists N_2 > 0 : \forall n \geq N_2 \mapsto \sum_{k=0}^{m}\frac{\frac{n^3}{(\sqrt{3})^k}}{\log\left(\frac{n}{(\sqrt{3})^k}\right)} \leq C_2\frac{n^3}{\log n}$$

Шаг индукции доказан.

Таким образом, $O(\frac{n^3}{\log n})$ оценка сверху, следовательно ответ:

$$T(n) = \Theta\left(\frac{n^3}{\log n}\right)$$