Домашнее задание № 3.
Алгоритмы и модели вычислений

Матвей Морозов, 678 группа

1 марта 2018 г.

Доп. задача 1

Найдите Θ-асимптотику реккурентной последовательности $T(n)$:

$$T(n) = T(\alpha n) + T((1-\alpha)n) + \Theta(n), \quad 0 < \alpha < 1$$

Считать, что $T(n)$ ограничено константой при достаточно малых n.

Решение:

1. Воспользуемся методом Akra–Bazzi.

   Решим уравнение $(1-\alpha)^p + \alpha^p = 1$
   $p = 1$

2. $$T(n) = \Theta(n^p(1 + \int_1^n \frac{du}{u})) = \Theta(n \log n)$$

Задача 1

1) Очевидно (по принципу Дирихле), что если мы берём три любые вершины двудольного графа, то хотя бы две из них будут в одной доле. По условию задачи нам нужно 2018 треугольников (т.е. три вершины, связанные между собой попарно), но по определению двудольных графов, вершины внутри каждой доли не связаны, поэтому этот язык пустой и распознаётся за $T = O(1)$, т.е. $\in P$

2) Известный факт: поиск циклов в графе удобно организовывать с помощью поиска в глубину (DFS), который работает за $O(V+E)$. Если во время DFS мы натыкаемся на уже отмеченную вершину - цикл найден.
Получается, что если наш граф задан матрицей смежности, то длина входа $n^2 \equiv V^2$. И т.к. число рёбер ограничено $0 \leq E \leq (V-1)^2$ (для полного графа). Хотя можно доказать, что в итоге этот алгоритм будет $O(n)$(при отсутствии циклов граф будет более разреженным), нам достаточно и оценки сверху: $O(n + n^2) = O(n^2)$. Таким образом

ассимптотика поиска цикла $T(n) < O(n^2)$ и язык полиномиален.

3) Самый грубый способ проверки без каких либо оптимизаций: будем просто из каждой ячейки матрицы с координатами $i \in \overline{1,2018}$ и $j \in \overline{1,2018}$ проходиться по всем элементам квадратной подматрицы от i,j до $i+n-2018, j+n-2018$ и прерываться при нахождении хотя бы одного 0. Такой алгоритм будет работать в худшем случае ассимптотически за $O(n^4)$, что доказывает его принадлежность P.

Задача 4

```
N=|v|;
ends={0};
for (i=1; i<=N; i++) {
        for (j in ends)
                if (MT(v[j+1...i])){
                        if(i==n) return true;
                        ends.include(i)
                }
        }
}
return false;
```

Суть алгоритма - фактически за $O(n^2)$ мы перебираем возможные разбиения v на подслова (индексы, по которым разбиваем хранятся в ends). Останавливаемся, когда слово удалось непрерывно (без непринадлежащих языку подслов) разбить. MT(v[j+1...i]) значит проверку на принадлежность слова языку. Оно полиномиально т.к. сам язык по предположению полиномиален. Тогда мы просто повышаем степень полинома на два, и получаем полином. Итоговый язык $L^* \in P$

Задача 2

Язык $L(a,m)$ состоит не более чем из m элементов. Все его элементы мы можем определить за $O(m)$. Принадлежность x к множеству элементов $L(a,m)$ можно проверить за $O(m)$. Значит, алгоритм полиномиален.