АиСД Листок №1.
P и NP

Антюх Михаил группа 176

21 сентября 2018 г.

Задача 1: Опишите одноленточную и двуленточную машины Тьюринга, разрешающие язык, состоящий из палиндромов (слов, читающихся одинаково слева направо и справа налево) в алфавите 0, 1. Оцените время работы предложенных машин.

Обозначения: λ - пустой символ

Одноленточная машина тьюринга:

Неформально:

На ленте записано входное слово, пусть головка изначально указывает на первый символ входного слова. Если на ленте изначально пусто (головка указывает на λ), то сразу переходим из q_{start} в q_{accept}. Если на ленте не пусто, то из q_{start} мы переходим в состояние q_0 если головка указывает на 0 или в q_1 если головка указывает на 1 и стираем символ на который указывала головка. В таком состояние мы идем по ленте пока головка не окажется на λ, делаем шаг влево, у нас есть три варианта:

- если символ совпадает с текущим состоянием, то стираем его, переходим в q_{return} и движемся в таком состояние влево до пустого символа, затем делает шаг вправо (головка оказывается на новом первом символе) и переходим в q_{start}. Повторяем все действия с самого начала.

- если символ не совпадает с текущим состоянием, то сразу переходим в $q_{rejected}$ и завершаем работу.

- если головка указывает на λ, то сразу перехлдим в q_{accept} и завершаем работу.

Формально:

	0	1	λ
q_{start}	(λ, q_0,$\rightarrow$)	(λ, q_1,$\rightarrow$)	(λ, q_{accept},$\rightarrow$)
q_0	(0, q_0,$\rightarrow$)	(1, q_0,$\rightarrow$)	(λ, $q_{0_{check}}$,$\leftarrow$)
q_1	(0, q_1,$\rightarrow$)	(1, q_1,$\rightarrow$)	(λ, $q_{1_{check}}$,$\leftarrow$)
$q_{0_{check}}$	(λ, q_{return},$\leftarrow$)	(1, q_{reject},$\leftarrow$)	(λ, q_{accept},$\leftarrow$)
$q_{1_{check}}$	(0, q_{reject},$\leftarrow$)	(λ, q_{return},$\leftarrow$)	(λ, q_{accept},$\leftarrow$)
q_{return}	(0, q_{return},$\leftarrow$)	(1, q_{return},$\leftarrow$)	(λ, q_{start},$\rightarrow$)

Двуленточная машина тьюринга:

Неформально:

На первой ленте записано входное слово, пусть головка первой ленты указывает на первый символ входного слова. Если головка первой ленты указывает на λ, то сразу переходим в q_{accept}. Если входное слово не пусто, то переходим в $q_{go_to_end}$ и идем в этом состояние до конца слова (головка второй ленты ничего не делает). Дойдя до конца слова, ставим головку на последний символ и переходим в q_{write}. В состояние $qwrite$ мы идем в начало слова параллельно записывая прочитанные символы на вторую ленту (головка второй ленты движется вправо). В итоге у нас на первой линте записано исходное слово, а на второй инвертированное. Возвращаем головку второй ленты на первый символ. Теперь просто идем по слову на двух лентах и если символы на двух лентах совпадают, то продолжаем движение, иначе сразу переходим в q_{reject} и завершаем работу. В процессе работы мы либо перейдем в q_{reject} либо дойдем до λ. В случаем, когда мы доходим до λ, то переходим в q_{accept}.

Время работы:

Двуленточная машина совершает два прохода по входному слову длинны n, когда переносит его инверсию на вторую ленту. И еще один проход, когда сравнивает слова на первой и второй ленте. Время работы $= O(n)$.
Одноленточная машина совершает в начале проход по слову длинны n, затем 2 раза по слову длинны $n-1, \ldots, 1$.
$n + 2(n-1) + \cdots + 2 = n^2$. Время работы $= O(n^2)$.

Задача 2: Покажите, что класс NP замкнут относительно пересечения и относительно операции *, т.е. если $L1 \in NP$ и $L2 \in NP$, то $L1 \cap L2 \in NP$ и $L^* \in NP$.

Доказательство. $L1 \in NP$ и $L2 \in NP$ => мы можем на машине Тьюринга за время, не превосходящее полинома от размера входных данных, проверить принадлежит ли слово языку L1 или L2 => мы можем за полином проверить принадлежил ли слово $L1 \cap L2$ => $L1 \cap L2 \in NP$
Для замыкания строим НМТ которая для входного слова переходит в состояния удовлетворяющие всем возможным разбиениям входного слова на подслова. Каждая ветвь

представляет собой некоторое разбиение на подслова и в каждой ветве имитируется работа машины тьюринга которая проверяет принадлежит ли текущее подслово языку L и так для каждого подслова в текущем разбиение. Если в какой-то из ветвей машина для каждого подслова выдаст accept, то это означает, что входное слово принадлежит L^*. Длины подслов <= n (n - длина входного слова), и количество подслов также <= n. Пусть для проверки каждого слова на исходной машине требуется $O(n^k)$, то каждая ветвь отработает за время = $O(n^{k+1})$ => $L^* \in NP$. □

Задача 3: Докажите, что следующая задача входит в класс P:
Дано: Две перестановки p и q множества $\{1, 2, \ldots, k\}$ и натуральное число t в двоичной кодировке.
Вопрос: Верно ли, что $p = q^t$ (где qt — это композиция t перестановок q)?

Доказательство. Для начала построим машину Тьюринга, которая складывает два десятичных числа. Строить такую машину формально очень долго и нудно, поэтому опишем ее очень не формально. Пусть наша машина трилентоточная, на первой ленте записано первое число, на второй ленте второе, а на третьей ленте мы будем писать результат от сложения двух чисел. Смотрим на последнюю цифру первого и второго числа, с помощью состояний определяем какая цифра получится при сложение и пишем ее на третью ленту, также с помощью состояний смотрим, надо ли прибавить единицу к следующему рязряду или нет. Двигаемся на первых двух лентах к следующим цифрам, складываем их, смотрим надо ли полученное число увеличить на единицу, и также проверяем переходит ли единица на следующие разряды. Пишем полученную цифру на третью ленту, и так далее. В результате на 3 ленте перевернутое число в десятичной системе. Переворачиваем его. Данная машина работает за $O(n)$.

Теперь машину, которая переводит число из двоичной в десятичную. Пусть у нас есть три ленты, на первой исходное число в двоичной системе, на второй записана 1, на третьей будет 0. Идем на первой ленте с конца числа, если видим единицу, то складываем с помощью первой машины 1 столько раз, сколько у нас записано на второй ленте, прибавляем результат к тому, что записано на третьей ленте, затем складываем число на второй ленте с собой и перезаисываем его на вторую ленту. Идем дальше по числу на первой ленте и повторяем все тоже самое. Данная машина n раз вызывает первую машину, значит время работы примерно $O(n^2)$.

Машина которая представляет перестановку в виде произведения циклов, пусть на первой ленте записаны k чисел, на второй куда они переходят, на третьей будем писать циклы. Машина совершает переходы по перестановке и пишет их на вторую ленту, как только с помощью состояний мы поняли что первый цикл завершился, мы стираем на первых двух лентах числа которые входят в цикл и начинаем поиск следующего цикла. Время работы $O(n^2)$.

Теперь сделаем машину тьюринга которая возводит цикл в степень $p = tmod$ длина цикла. Пусть на первой ленте записан цикл, головка идет по нему с помощью состояний на вторую ленту записывает кто куда переходит, затем стирает слово на первой ленте и со второй переписывает результат на 1, с помощью состояний повторяем это p раз. Время работы $O(n^2)$. Циклов <= n => время работы для возведения всех циклов $O(n^3)$. Очевидно что все эти машины мы можем заменить одной большой машиной, которая

имитирует все выше описанные машины с помощью большого количества состояний. Общее время работы $O(n^3)$. □

Задача 4: Пусть существует алгоритм A(G,k) с полиномиальным временем работы, позволяющий определить, есть ли в графе G клика раз- мера k. Покажите, что в этом случае существует алгоритм B(G) с полиномиальным временем работы, находящий клику максималь- ного размера в графе G (именно саму клику, а не только ее размер).

Для начала проверим сущетсвует ли клика размера k с помощью алгоритма A, если нет, то завершаем работу. Иначе запускаем наш алгоритм B. Пусть наш алгоритм B в начале нумерует вершины графа, затем он удаляет одну из вершин графа со всеми инцидентными ребрами и передает полученный граф алгоритму A, если тот говорит нам, что клики размера к больше нету, то добавляем нашу вершину в массив, иначе возвращаем нашу вершину со всеми ребрами и переходим к другой вершине, удаляем ее и так далее. В результате получим массив с вершинами которые входят в клику. Осталось только добавить ребра которые связывают эти вершины и передать в ответ полученный результат. Пусть алгоритм A работает за $O(n^k)$, алгоритм B тратит какую-то константу на нумерацию, удаление ребер и т.д и запускает n раз алгоритм A => время работы = $O(n^{k+1})$

Задача 5: Пусть существует алгоритм C(G,k) с полиномиальным временем работы, позволяющий определить, можно ли раскрасить вершины графа G в k цветов так, что никакие две смежные вершины не окажутся окрашенными в один цвет. Покажите, что в этом случае существует алгоритм D(G, k) с полиномиальным временем работы, предъявляющий раскраску графа G в k цветов, удовлетворяющую указанному выше условию, если это в принципе возможно, и сообщающий о том, что это невозможно в противном случае.

Для начала запускаем алгоритм C(G,k), который проверяет является ли граф к-раскрашиваемым, если нет, то сообщаем об этом и завершаем работу. Иначе строим дополнительный граф G'. G' - полный граф на k вершинах (очевидно, что G' к-раскрашиваемый). Пусть вершины графа G и G' пронумерованы. Наш алгоритм D будет брать по порядку вершины из графа G и соединять их k-1 ребром с k-1 вершиной из G', после присоединения запускаем на полученном графе алгоритм C, если он выдал, что граф к-раскрашиваемый, то записываем, что мы красим нашу вершинку в цвет номер i (i номер той вершины в G' которую мы не соединили с текущей вершиной из G, иначе соединяем текущую вершину из G с другой k-1 вершиной из G' и запускаем снова алгоритм C. В результате у нас каждой вершине в G будет сопоставлена одна вершина из G', номер которой будет означать цвет. Время работы: для каждой вершины нам в худшем случае потребуется k раз запустить алгоритм C, всего вершин n => в худшем случае потребуется kn раз запустить алгоритм C.