Задание 3

Автоматы и распознавание текстов; Лемма о накачке

Ключевые слова [1]:*принцип мат. индукции, язык, регулярные выражения, конкатенация, объединение, итерация, конечные автоматы (КА), детерминированные и недетерминированные КА, регулярные языки. алгебра регулярных выражений, примеры нерегулярных языков; поиск подстрок, алгоритм Кнута- Морисса- Пратта.*

1 НКА и ДКА

Из алгоритма детерминизации НКА следует, что если НКА $\mathcal{A}$ имеет множество состояний $Q_{\mathcal{A}}$, то построенный по нему ДКА $\mathcal{B}$ имеет множество макросостояний $Q_{\mathcal{B}} \subseteq 2^{Q_{\mathcal{A}}}$, где $2^{Q_{\mathcal{A}}}$ – множество всех подмножеств множества $Q_{\mathcal{A}}$. Таким образом, на число состояний автомата $\mathcal{B}$ мы имеем верхнюю оценку $|Q_{\mathcal{B}}| \leqslant 2^{|Q_{\mathcal{A}}|}$. То есть число состояний в ДКА ограниченно экспоненциальной функцией от числа состояний в НКА, но существует ли язык, для которого эта оценка достигается? На самом деле, когда мы говорим об оценках такого рода, нам требуется рассматривать ни один какой-то язык, а последовательность языков, по которым мы и сможем установить экспоненциальную зависимость.

Задача 1. Определим язык $L_i = \{w \,|\, |w| = n, w[n-i] = 1\}$, то есть в язык L_i входят все слова, в которых 1 стоит на i-ом месте от конца[2]. Постройте НКА, распознающий язык L_3. По построенному НКА постройте ДКА.

Задача 2*. Докажите, что на языках L_i между НКА и построенными по ним ДКА достигается экспоненциальный разрыв.

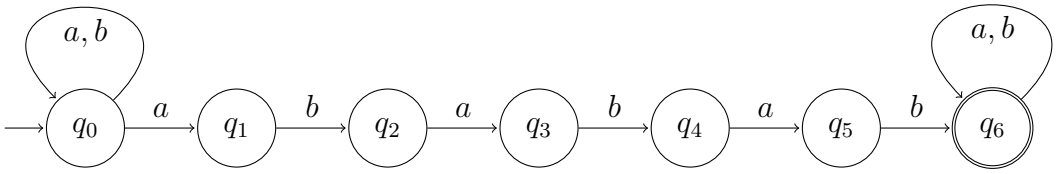
Упражнение 1. Почему для доказательство экспоненциального разрыва необходима бесконечная последовательность языков, а не достаточно конечной?

[1] минимальный необходимый объём понятий и навыков по этому разделу)

[2] Во избежании путаницы, первый с конца символ – это последний символ слова.

2 Алгоритм Кнута-Морриса-Пратта и его связь с автоматами

НКА – очень удобный инструмент для описания автоматов, которые ищут слова в тексте. Например, автомат



проверяет имеет ли поданное на вход слово подслово $ababab$.

Как мы уже обсуждали, для алгоритмической проверки принадлежности слова языку, распознаваемому НКА, по нему следует строить ДКА. Однако, в специальных случаях, используемых на практике, подобно описанному выше, есть более удобные алгоритмы и один из них – алгоритм Кнута-Морриса-Пратта. Этот алгоритм подробно описан в 10-ой главе книги А. Шеня «Программирование. Теоремы и задачи». Её можно в свободном доступе скачать здесь. Я рекомендую изучить КМП-алгоритм по этой книге, в этом разделе я лишь скажу пару слов о его связи с автоматами, а точнее дам на эту тему пару задач.

В основе этого алгоритма – использование для поиска слова вычисления префикс-функции.

Определение 1. Назовём префикс-функцией функцию $l()$, которая возвращает самый длинный несобственный[3] префикс слова w, являющийся одновременно его суффиксом.

Пример 1. Приведём пример вычисления префикс-функции.

$$l(a^{n+1}) = a^n$$
$$l(ababa) = aba$$
$$l(abb) = \varepsilon$$

[3]То есть префикс, не совпадающий со всем словом w.

У префикс функции есть важное свойство – все несобственные префиксы слова w, которые являются его суффиксами лежат в последовательности $l(w), l(l(w)), \ldots$

Задача 3*. Докажите, что в ДКА, распознающем язык $\Sigma^* w \Sigma^*$ не может быть меньше состояний чем элементов последовательности $l(w), l(l(w)), \ldots$

На семинаре мы разобрали алгоритм построения КМП-автомата. Он основывался на том, что КМП-автомат, который ищет в тексте t (вход автомата) слово w (фиксированный параметр автомата) имеет $|w|+1$ состояние, каждое из которых кодирует некоторый префикс слова w. Если автомат находится в i-ом состоянии , то это означает, что обработанный участок текста имеет вид

$$t_1 t_2 \cdot \ldots \cdot t_k w_1 w_2 \cdot \ldots \cdot w_i$$

Мы считаем, что нулевое состояние кодирует пустой префикс, а значит $w_0 = \varepsilon$. При этом, i – максимальная длина суффикса текста, совпадающая с префиксом w. Таким образом, если следующий символ текста не совпадает с символом w_{i+1}, то тогда необходимо найти новый максимальный по длине суффикс текста, который является префиксом w и сделать переход по отличающейся букве в состояние, соответствующее этому префиксу. При этом, оказывается, что его длина будет обязательно не превосходить i.

Задача 4. Почему?

Поскольку w имеет конечное число префиксов, то вычислить переходы КМП-автомата можно на конечной памяти полным перебором безо всякой науки. Но это можно сделать и используя высокую науку, обсуждение которой я был вынужден вероломно прервать на семинаре. В какой-то момент я сформулировал и объяснил утверждение из следующей задачи, потом часть аудитории его успела забыть, и в итоге не осталось времени с ним разбираться. Тем не менее, это утверждение верное и позволяет ускорить вычисление переходов КМП-автомата.

Задача 5*. Пусть $l(xay) = x$, где x, y слова над алфавитом $\{a, b\}$. Тогда $l(xayb) = l(xb)$. Докажите это утверждение.

Результаты предыдущих задач позволяют говорить о том, что КМП-алгоритм можно представить в виде последовательного вычисление префикс-функции на словах вида

$$w\#t_1 \cdot \ldots \cdot t_n$$

При этом, вычисление $l(w\#t_1 \cdot \ldots \cdot t_n t_{n+1})$ по значению $l(w\#t_1 \cdot \ldots \cdot t_n)$ осуществляется за константное время, то есть ничего не стоит.

3 Лемма о накачке[4]

В данном разделе мы поговорим о лемме о накачке – одном из способов доказательства нерегулярности языка.

Лемма 1. *Для любого регулярного языка L существует такая константа $p \geqslant 1$, что для любого слова w из L длиннее p, справедливо:*

- $w = xyz$
- $|y| \geqslant 1$
- $|xy| \leqslant p$
- $\forall i \geqslant 0, xy^i z \in L$.

Доказательство. Поскольку $L \in \mathsf{REG}$, то существует ДКА $\mathcal{A}$ распознающий L. Пусть $\mathcal{A}$ имеет N состояний. Возьмём $p = N + 1$. Тогда, если если слово w принадлежит L и $|w| \geqslant p$, то это означает, что при обработке w автомат $\mathcal{A}$ оказался в некотором сотоянии q дважды. Пусть в первый раз автомат оказался в q после прочтения префикса x, а второй раз, при прочтении префикса xy. Тогда $\delta(q, y) = q$, но поскольку $w = xyz$ принадлежит L, то это означает, что $\delta(q, z) = q_f \in F$, а значит все слова вида $xy^i z$, $i \geqslant 0$ лежат в L. □

Обратите внимание, что при доказательстве леммы, я использовал те же трюки, что и в доказательстве на семинаре того, что $a^n b^n$ – нерегуляный язык. Также обратите внимание на то, что формула $w = xyz$ означает, что для слова w существует такое разбиение xyz, для которого выполняются следующие свойства и утверждение леммы: часто студенты это равенство ошибочно воспринимают как «для любого разбиения».

[4]Также известна как лемма о разрастании. Неудачные переводы неудачного термина «Pumping Lemma».

Пример 2. Используем лемму о накачке для доказательства христоматийного примера нерегулярности языка $L = \{0^n 1^n \mid n \geqslant 0\}$.

Доказательство. Допустим, что язык L регулярный. Тогда, по лемме о накачке, существует константа p, что для любого слова w длиннее p, существует такое разбиение xyz, что $|xy| \leqslant p$ и слова $xy^i z$, $i \geqslant 0$ принадлежат L.

Рассмотрим $w = 0^p 1^p$. Если такое разбиение существует, то y имеет вид 0^k или 1^k, $k \geqslant 1$ – в противном случае, если $y = 0^k 1^l$, то $y^2 = 0^k 1^l 0^k 1^l$, но в L нет слов, в которых за 1 следует 0. Допустим, что $y = 0^k$. Тогда $x = 0^m$, $z = 0^l 1^p$, $k + m + l = p$. Но тогда, по лемме о накачке $xy^2 z \in L$, а значит, слово $0^{m+2k+l} 1^l \in L$, но $m + 2k + l > p$, т.к. $m + k + l = p$ и $k > 0$, поскольку $|y| \geqslant 1$. Аналогично приходим к противоречию когда $y = 1^k$. □

Упражнение 2. В предыдущем примере показано громоздкое доказательство: его можно сделать проще, убрав перебор кучи случаев, воспользовавшись структурой слова. Постарайтесь получить доказательство в одну строчку.

У этой леммы слишком много минусов. Во-первых, она работает не всегда: если язык нерегулярен, это ещё не означает, что лемма о накачке для него не выполняется. Во-вторых, она слишком громоздкая. Даже для такого простого примера как $L = \{0^n 1^n\}$, неподкованному в этой науке человеку потребуется много писанины, а в более сложных случаев перебор возможных y куда шире. Как показывает наблюдение, руками нерегулярность доказать быстрее, да и работает техника, обсужденная на семинаре в тех случаях, когда применима лемма о накачке. Но тем не менее, у этой леммы есть и плюсы – учебные. Во-первых, лемма о накачке показывает структуру регулярного языка: разность длин двух последовательных слов из регулярного языка ограниченна линейной функцией. Во-вторых, сущетсвует ещё лемма о накачке для КС-языков, для понимания которой стоит изучить более простую лемму о накачке для регулярных языков. В случае КС-языков, доказательство непринадлежности языка классу КС уже куда менее очевидно, так что лемма о накачке становится мощным и одним из основных инструментов.

Задача 6. Применить лемму о накачке для доказательства нерегулярности языка $L = \{a^{2^n} \mid n \geqslant 0\}$.

4 Дополнительные задачи

Задача 7. Приведите протокол работы КМП-алгоритма при поиске подслова *abba* в слове *abbbababbab*.