Задание 3

НКА и ДКА
Алгоритмы обработки текстов I

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-Морриса-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 ДКА ограниченно экспоненциальной функцией от числа состояний 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 доказательства 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

Одно из простых и естественных применений регулярных выражений – поиск в тексте (слове) t вхождения некоторого слова w. В терминах регулярных выражений, задача состоит в проверке принадлежности слова t языку, порождаемому РВ $\Sigma^* w \Sigma^*$. Однако, сходу придумать алгоритм, который выполняет эту проверку за линейное время не очень просто. Таким алгоритмом является алгоритм Кнута-Морриса-Пратта (КМП) и здесь мы опишем его на языке автоматов. С самим алгоритмом можно познакомиться, например 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.

Для того, чтобы описать КМП-алгоритм, нам потребуется сначала ввести понятие префикс-функции.

Определение 1. Назовём *префикс-функцией* функцию $l : \Sigma^* \to \Sigma^*$, которая возвращает самый длинный собственный[3] префикс слова w, являющийся одновременно его суффиксом.

Пример 1. Приведём пример вычисления префикс-функции.

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

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

Упражнение 2. Докажите это свойство.

Зафиксируем слово w. Обозначим через $w[i, j]$ подслово $w_i w_{i+1} \ldots w_j$ слова $w = w_1 w_2 \ldots w_n$, здесь всюду $w_k \in \Sigma$. Для удобства будем считать, что $w[0, 0] = \varepsilon$, а $w[0, k] = w[1, k]$ при $k > 0$. Определим автомат, который будем называть *автоматом Кнута-Морриса-Пратта* или КМП-автоматом для слова w.

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

Определение 2. КМП-автоматом $\mathcal{A}_w$ для слова $w \in \Sigma^*$ длины n, называется автомат, который задан набором $(Q,\ \Sigma,\ q_0,\ \delta,\ F)$, где

- $Q = \{\ w[0,0],\ w[0,1],\ w[0,2], \ldots,\ w[0,n]\ \}$;
- $q_0 = w[0,0]$;
- $\delta(w[0,k],a) = \begin{cases} w[0,k+1], & \text{при } w[0,k+1] = w[0,k]a \text{ и } k < n; \\ l(w[0,k]a), & \text{при } w[0,k+1] \neq w[0,k]a \text{ и } k < n; \\ w[0,n], & \text{при } k = n. \end{cases}$
- $F = \{w[0,n]\}$.

Замечание 1. *В качестве множества состояний КМП-автомата выступает множество слов, поэтому применение для состояний операций со словами, например «$w[0,k+1] = w[0,k]a$», при определении функции переходов корректно и осмысленно.*

Упражнение 3. Убедитесь, что КМП-автомат для слова *bababa*, почти совпадает с ДКА, построенным на семинаре по НКА для $\Sigma^*bababa\Sigma^*$

Задача 3. Постройте КМП-автомат для слова *abababb* и продемонстрируйте его работу на слове *abababbabb*.

Алгоритм построения КМП-автомата кажется довольно простым. Однако, его ключевое место – вычисление префикс функции. Очевидное переборное вычисление префикс-функции осуществляется квадратичным алгоритмом (за $O(|w|^2)$). Оказывается, префикс-функцию специального вида, $l(w[0,k]a), w[k+1] \neq a$, можно вычислить линейным алгоритмом! Примерно это и реализует оригинальный КМП-алгоритм, который работает за линейное время. В книжке Шеня представлен модифицированный вариант КМП-алгоритма, который чуть проще реализовать; он также работает за линейное время – оценка следует из амортизационного анализа (см. задачу 10.4.3 этой).

3 Структура данных «Словарь»

Под словарём понимают структуру данных, с помощью которой можно проверять вхождение слова в множество. У этой структуры данных есть следующие операции:

- **in**: добавить слово x в словарь;
- **test**: проверить входит ли слово x в словарь;
- **out**: удалить слово x из словаря.

Эту структуру данных можно реализовать с помощью автомата. На рис. 1 показан пример словаря с содержимым $S = \{a, ba, ab, abb, abab\}$, который реализован через ДКА. ДКА $\mathcal{A}$ принимает слово x тогда и только тогда, когда $x \in S$.

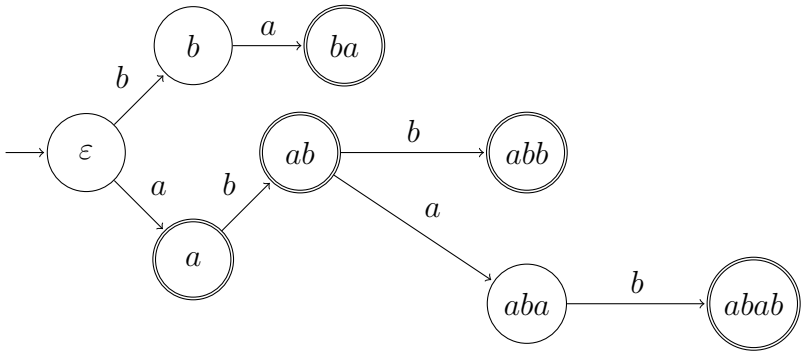

Рис. 1: ДКА $\mathcal{A}$ реализует словарь

Задача 4. Постройте алгоритмы, реализующие операции **in** и **out** для ДКА, реализующих словарь. В результате этих операций должен получиться ДКА, который реализует словарь с соответственно изменившимся содержимым.

Задача 5. Постройте алгоритм, который получает на вход конечное множество слов S и возвращает ДКА, реализующий словарь с содержимым S. Используйте при построении только известные из курса алгоритмы для регулярных множеств (выражений) и конечных автоматов.

Замечание. В задачах, в которых требуется построить алгоритм, необходимо доказательство его корректности.

Задача 6. Постройте ДКА для словаря $\{ac, acb, b, ba, c, cbb\}$. Добавьте в полученный словарь слово ab и удалите слово ac.

4 Дополнительные задачи

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

Замечание 2. *Здесь идёт речь именно о КМП-алгоритме, а не о демонстрации работы КМП-автомата! Кроме того, префикс функция должна быть вычислена по алгоритму, описанному в книжке Шеня, а не методом внимательного вглядывания.*