КИБЕРНЕТИКА

К.В. ШАХБАЗЯН, Ю.Г. ШУКУРЯН

УДК 519.6

АСИНХРОННЫЕ АВТОМАТЫ, СРАВНИВАЮЩИЕ ТРЕКИ

Ключевые слова: *асинхронный автомат, трек, сравнение.*

Рассмотрим алгоритмы решения нескольких задач сравнения треков Мазуркевича [1, 2], которые сводятся к построению автоматов, распознающих соответствующие рациональные трековые языки. Эти языки и их свойства исследовались многими авторами [1–7]. Ниже приведены трековые языки, связанные с решением конкретных задач, имеющих аналоги в науке о строках (stringology) [8–12]. Поскольку все описанные языки рациональные, для них заведомо существуют допускающие их автоматы Зеленки. Асинхронные автоматы Зеленки [3] представляют собой множество последовательных компонентов, взаимодействие между которыми моделируется следующим синхронизационным ограничением: компонент может выполнять переход по некоторой букве тогда и только тогда, когда одновременно выполняют переход по той же букве все компоненты асинхронного автомата, имеющие ее в своем алфавите.

Интерес вызывают вопросы о сложности алгоритмов построения автоматов, сравнивающих треки, и о сложности сравнения треков.

Известно, что для каждого рационального языка строк L_{str}, распознаваемого автоматом A_{str}, существует алгоритм $\mathcal{U}_{\text{tr}}$, распознающий трековый язык $L_{\text{tr}} = [L_{\text{str}}] = \{x \in \Sigma^* \mid \exists\, y \in L_{\text{str}} \Rightarrow y \approx_I x\}$. Как показано в [7], для языка L_{tr} в общем случае алгоритм сводится к применению автомата A_{str} ко всем префиксам входной строки и имеет сложность в худшем случае $O(|x|)^{\alpha}$, где α — число вершин максимальной клики графа независимости (Σ, I).

Ниже изложены методы построения асинхронных автоматов для решения следующих задач:

- распознавание вложенных треков — по заданной строке $y \in \Sigma^*$ строится автомат, распознающий треки, вложенные в трек $[y]$;
- распознавание конечного языка треков — по заданным строкам $x_1, \dots, x_n \in \Sigma^*$ строится автомат, распознающий трековый язык $[x_1] \cup \dots \cup [x_n]$;
- распознавание суффиксов трека — по заданной строке $x \in \Sigma^*$ строится автомат, распознающий язык $\mathrm{Suff}([x])$;
- по «словарю» — множеству строк $\{x_1, \dots, x_n\}$, строится автомат, распознающий язык $[\Sigma^*]([x_1] \cup \dots \cup [x_n])$;

• по «словарю» — множеству строк $\{x_1, \ldots, x_n\}$, строится автомат, решающий задачу поиска в строке $y = y_1, \ldots, y_{|y|} \in \Sigma^*$ всех префиксов $y_1, \ldots, y_k$ строки y, для которых существует x_j такое, что $[x_j] \in \mathrm{Suff}[y_1 \ldots y_k]$.

Для решения перечисленных задач предложены алгоритмы с линейной временной сложностью. Приведены оценки сложности построения соответствующих асинхронных автоматов и задержки на каждом шагу их работы.

Все автоматы, описанные в статье, относятся к классу суперпозиций распределенных асинхронных автоматов и автоматов, распознающих конечные языки строк.

ПРЕДВАРИТЕЛЬНЫЕ СВЕДЕНИЯ И ОПРЕДЕЛЕНИЯ

Пусть Σ — конечный алфавит, $D \subset \Sigma \times \Sigma$ — рефлексивное и симметричное отношение зависимости, $I = (\Sigma \times \Sigma) \setminus D$ — отношение независимости или перестановочности. Отношение I индуцирует отношение эквивалентности $\approx_I$ на Σ^*. Две строки $x, y \in \Sigma^*$ эквивалентны относительно $\approx_I$, если существует последовательность $z_1, \ldots, z_n$ строк таких, что $x = z_1$, $y = z_n$, и для всех i $(1 \le i < n)$ существуют строки $z_i' z_i''$ и буквы a_i, b_i, удовлетворяющие условиям

$$\begin{cases} z_i = z_i' a_i b_i z_i'', \\ z_{i+1} = z_i' b_i a_i z_i'', \text{ где } (a_i, b_i) \in I, \end{cases}$$

т.е. две строки эквивалентны по отношению $\approx_I$ тогда и только тогда, когда одну из них можно получить из другой путем перестановок соседних независимых букв. Тогда множество $M = M(\Sigma, D)$ классов эквивалентных по отношению $\approx_I$ строк в алфавите Σ называется трековым моноидом, а его элементы — треками. На элементах M определено умножение — конкатенация. Трек t обозначается $[x]$ для любой представляющей строки $x \in t$. Длина трека t есть длина любого его представителя $y \in t$ и обозначается $|t| = |y|$.

Пусть $(\Sigma_1, \ldots, \Sigma_m)$ — покрытие алфавита зависимости (Σ, D) кликами, т.е. семейство подмножеств Σ таких, что

$$\bigcup_{i=1}^{m} \Sigma_i = \Sigma, \ \ \Sigma_i \times \Sigma_i \subseteq D \ \ (i = 1, 2, \ldots, m),$$

$$(a, b) \in D \Leftrightarrow \exists\, i : a, b \in \Sigma_i .$$

Далее везде будем считать фиксированными трековый моноид $M(\Sigma, D)$ и $\Sigma_1, \ldots, \Sigma_m$ — наименьшее покрытие кликами алфавита зависимости (Σ, D). Тогда m — число клик в наименьшем покрытии.

Любой трек $t \in M(\Sigma, D)$ можно представить m-й строкой [1], которую обозначим $\pi(t) = \{\pi_1(t), \ldots, \pi_m(t)\}$. Здесь $\pi_i(t) \in \Sigma_i^*$ — проекция строки $y \in t$ на Σ_i.

Пусть $s, t \in M(\Sigma, D)$. Трек s вложен в трек t, если существуют треки $t_0, \ldots, t_l, s_1, \ldots, s_l \in M(\Sigma, D)$ такие, что $t = t_0 s_1 t_1 \ldots s_l t_l$ и $s = s_1 \ldots s_l$ [13].

Для заданного трека $t \in M(\Sigma, D)$ говорят, что p есть префикс t и q есть суффикс t, если $t = pq$, где $p, q \in M(\Sigma, D)$. Обозначим $\mathrm{Pref}(t)$ множество префиксов трека t и $\mathrm{Suff}(t)$ — множество суффиксов трека t.

Определение 1. $\mathcal{P}$-автоматом (product automaton) над моноидом $M(\Sigma, D)$ называется структура $B = (A_1, \ldots, A_m, \Phi)$. Здесь $A_1, \ldots, A_m$ — детерминированные конечные автоматы над строками $A_i = (\Sigma_i, Q_i, q_i^0, \delta_i, F_i)$, $i = 1, \ldots, m$, называемые локальными компонентами, где Q_i — множество внутренних состояний, δ_i —

частично определенная функция перехода, F_i — множество финальных состояний автомата A_i, $\Phi \subseteq F_1 \times \ldots \times F_m$ — множество глобальных финальных состояний автомата B. Множество $Q = Q_1 \times \ldots \times Q_m$ называется множеством глобальных состояний.

Если считать локальные компоненты линейно упорядоченными, то множество $\Phi \subseteq F_1 \times \ldots \times F_m$ можно рассматривать как конечный язык финальных состояний и, в частности, можно говорить об автомате A_Φ, распознающем принадлежность строк языку Φ.

Правила переходов между глобальными состояниями задают синхронизационные ограничения между локальными компонентами $\mathcal{B}$-автомата, сформулированные в следующем определении.

Определение 2. Глобальный переход δ есть частично определенная функция $\delta : Q \times \Sigma \to Q$, где значение $\delta((q_1, \ldots, q_m), a) = (q'_1, \ldots, q'_m)$ определено тогда и только тогда, когда определены локальные переходы $\delta_i(q_i, a)$ для всех $1 \le i \le m$ таких, что $a \in \Sigma_i$. Значение $((q_1, \ldots, q_m), a)$ вычисляется по формуле

$$q'_i = \begin{cases} q'_i = \delta_i(q_i, a), & \text{если } a \in \Sigma_i, \\ \quad q_i, & \text{если } a \notin \Sigma_i. \end{cases}$$

Определение 3. Пробегом $\mathcal{P}$-автомата $B = (A_1, \ldots, A_m, \Phi)$ на треке t, заданном строкой $y \in t$, называется отображение $\rho : \mathrm{Pref}(y) \to Q$ такое, что $\rho(\varepsilon) = (q_1^0, \ldots, q_m^0)$ и для всех $a \in \Sigma$, $\tau \in \mathrm{Pref}(y) : \rho(\tau a) = \delta(\rho(\tau), a)$, если глобальный переход определен. Очевидно, что если строки $y \approx_I y'$, то $\rho(y) = \rho(y')$. Говорят, что $\mathcal{P}$-автомат B допускает трек t, если $\rho(y)$ определено и $\rho(y) \in \Phi$.

Определение 4. $\mathcal{P}$-автоматы с универсальным финальным множеством $U = F_1 \times \ldots \times F_m$ в [6] называются дистрибутивными (распределенными) $\mathcal{B}$-автоматами.

Рассмотрим $\mathcal{P}$-автомат как суперпозицию распределенного автомата и автомата A_Φ, определяющего принадлежность достигнутого глобального состояния $(q_1, \ldots, q_m)$ языку финальных глобальных состояний. Поскольку каждое глобальное состояние $(q_1, \ldots, q_m)$ можно рассматривать как строку $q_1 \ldots q_m$ в алфавите $Q_1 \times \ldots \times Q_m$, то A_Φ есть автомат, допускающий строки языка финальных состояний Φ.

Будем полагать, что каждый глобальный переход $\delta((q_1, \ldots, q_m), a) = (q'_1, \ldots, q'_m)$ распределенного $\mathcal{P}$-автомата B сопровождается выяснением принадлежности достигнутого состояния множеству финальных состояний, т.е. применением автомата A_Φ к достигнутому состоянию $(q'_1, \ldots, q'_m)$.

Уточним понятие $\mathcal{P}$-автомата.

Определение 5. $\mathcal{P}$-автомат есть либо распределенный $\mathcal{P}$-автомат $(A_1, \ldots, A_m, U)$, либо суперпозиция распределенного $\mathcal{P}$-автомата $(A_1, \ldots, A_m, U)$ и автомата A_Φ над алфавитом $Q_1 \times \ldots \times Q_m$, допускающим язык строк $\Phi = \{q_1 \ldots q_m \mid (q_1, \ldots, q_m) \in \Phi\}$.

Определение 6. Пусть $B = (A_1, \ldots, A_m, \Phi)$ — произвольный автомат. Будем называть задержкой автомата B суммарную сложность вычисления функции перехода δ и вычисления предиката $(q_1, \ldots, q_m) \in \Phi$. Обозначим $Z(A_\Phi)$ задержку автомата A_Φ.

Для произвольного автомата A задержка $Z(A)$ есть тот минимальный интервал времени, который должен разделять последовательные символы, поступающие на вход автомата A, т.е. входная строка y допускается автоматом A за время $|y| Z(A)$.

Для $\mathcal{P}$-автомата $B = F(A_1, \dots, A_m, \Phi)$ при последовательной реализации имеет место $Z(B) = Z(A_\Phi) + \Sigma_1^m Z(A_i)$, а при параллельной реализации очевидно, что $Z(B) = Z(A_\Phi) + \max_i Z(A_i)$. Ниже все оценки задержек $Z(B)$ приведены для последовательной реализации.

Утверждение 1. Пусть заданы автоматы $A_i = (\Sigma_i, Q_i, q_i^0, \delta_i, F_i)$, $i = 1, \dots, m$, распознающие языки строк $L(A_i)$. Распределенный $\mathcal{P}$-автомат $B = (A_1, \dots, A_m, U)$ распознает язык $L(B)$ тех и только тех треков $t \in M(\Sigma, D)$, для которых $\pi_i(t) \in L(A_i)$, где $\pi_i(t)$ — проекция трека t на алфавит Σ_i. Если все автоматы A_i минимальны и $(\Sigma_1, \dots, \Sigma_m)$ — наименьшее покрытие кликами алфавита зависимости (Σ, D), то автомат B также минимален. Задержка автомата с универсальным финальным множеством есть $Z(B) = \Sigma_1^m Z(A_i)$.

Доказательство. Пусть на вход автомата B поступает строка y. Согласно определению на вход локального компонента A_i $(i = 1, \dots, m)$ попадет подпоследовательность строки y, все символы которой принадлежат Σ_i, т.е. проекция $\pi_i(y)$. Отсюда следует, что если строка допускается автоматом B, то и каждая строка, входящая в трек $[y]$, тоже допускается им, так как трек однозначно определяется набором проекций любого его представителя.

РАСПОЗНАВАНИЕ ВЛОЖЕННЫХ ТРЕКОВ

Приведем примеры языков, распознаваемых с помощью распределенных $\mathcal{P}$-автоматов.

Задача 1. Задана строка y. Построить $\mathcal{P}$-автомат $S_{\text{tr}}(y)$, распознающий язык треков $L_{\text{tr}}(y) = \{t \in M(\Sigma, D) \mid t \subset [y]\}$, вложенных в трек $[y]$.

Замечание. Далее нижние индексы tr и st используются соответственно для языков треков и строк и автоматов, распознающих эти языки.

Утверждение 2. Язык $L_{\text{tr}}(y)$ распознается распределенным $\mathcal{P}$-автоматом $S_{\text{tr}}(y) = (S_{\text{st}}(\pi_1(y)), \dots, S_{\text{st}}(\pi_m(y)), U)$, где автомат $S_{\text{st}}(\pi_i(y))$ распознает подпоследовательности проекции $\pi_i(y)$. Автомат $S_{\text{tr}}(y)$ можно построить за время $O(m|y| \times \log|y|)$. Он имеет задержку $O(m)$.

Доказательство. Известно, что автомат $S_{\text{st}}(y) = \text{DASG}(y)$ [8] распознает язык $L_{\text{st}}(y)$ всех строк, являющихся подпоследовательностями строки y. Его можно построить за время $O(|y| \times \log|y|)$. Воспользуемся автоматом $S_{\text{st}}(y)$ для строк при построении $\mathcal{P}$-автомата $S_{\text{tr}}(y)$ для треков. Из определения вложенности треков следует, что для того чтобы трек $t \in M(\Sigma, D)$ был вложен в трек $[y]$, необходимо и достаточно, чтобы для всех $i \in \{1, \dots, m\}$ было выполнено $\pi_i(x) \in L_{\text{st}}(\pi_i(y))$, причем для всех $a \in \Sigma$ каждое j-е вхождение буквы a в $\pi_i(x)$ синхронизируется с j-м вхождением буквы a в $\pi_i(y)$, а это, в свою очередь, согласуется с синхронизационным ограничением для $\mathcal{P}$-автомата. Следовательно, распределенный $\mathcal{P}$-автомат $S_{\text{tr}}(y) = (S_{\text{st}}(\pi_1(y)), \dots, S_{\text{st}}(\pi_m(y)), U)$ распознает язык $L_{\text{tr}}(y)$. Сложность построения вытекает из оценки сложности построения автомата $S_{\text{st}}(y) = \text{DASG}(y)$. Автомат имеет задержку $Z(S_{\text{tr}}(y)) = O(m)$.

РАСПОЗНАВАНИЕ СУФФИКСОВ ТРЕКА

Задача 2. Задана строка y. Построить $\mathcal{P}$-автомат, распознающий язык $\text{Suff}([y])$ суффиксов трека $[y]$.

Задача построения автомата, распознающего суффиксы строки y, имеет известное решение в виде минимального автомата $\mathcal{M}_{\text{st}}(t)$ [9–11], который можно построить за время $O(|y|)$.

Утверждение 3. Пусть трек $t \in M(\Sigma, D)$ представлен строкой $y \in t$ и пусть $\mathcal{M}_{\mathrm{st}}(\pi_i(y))$, $i=1,\ldots,m$, — минимальные автоматы, распознающие языки $\mathrm{Suff}(\pi_i(y))$. Распределенный $\mathcal{P}$-автомат $\mathcal{M}_{\mathrm{tr}}(t) = (\mathcal{M}_{\mathrm{st}}(\pi_1(y)), \ldots, \mathcal{M}_{\mathrm{st}}(\pi_m(y)), U)$ распознает язык $\mathrm{Suff}([y])$ суффиксов трека $t=[y] \in M(\Sigma, D)$. Автомат $\mathcal{M}_{\mathrm{tr}}(t)$ можно построить за время $O(m|t|)$, при этом он имеет задержку $Z(\mathcal{M}_{\mathrm{tr}}(t)) = O(m)$.

Доказательство. Как известно [9–11], минимальный автомат $\mathcal{M}_{\mathrm{st}}(\mathrm{Suff}((y)))$, распознающий язык $\mathrm{Suff}(y)$ суффиксов строки y, можно построить за время $O(|y|)$, при этом он имеет задержку $O(1)$. Очевидно, что трек t' является суффиксом трека t тогда и только тогда, когда для всех $i=1,\ldots,m$ строка $\pi_i(t')$ является суффиксом строки $\pi_i(t)$. Отсюда следует, что автомат $\mathcal{M}_{\mathrm{tr}}(t)$ действительно распознает язык $\mathrm{Suff}(t)$ суффиксов трека t. Очевидно, что автомат $\mathcal{M}_{\mathrm{tr}}(t)$ можно построить за время $O(m|y|)$ и $Z(\mathcal{M}_{\mathrm{tr}}(t)) = O(m)$.

РАСПОЗНАВАНИЕ КОНЕЧНОГО ЯЗЫКА ТРЕКОВ

Множество строк $\{x_1, \ldots, x_n\} \in \Sigma^*$ можно, с одной стороны, рассматривать как элементы свободной полугруппы, а с другой — как множество представителей треков $t_1 = [x_1], \ldots, t_n = [x_n]$. Множество строк обозначим $X = \{x_1, \ldots, x_n\}$, а множество соответствующих треков $[X] = \{[x] \mid x \in X\}$.

Задача 3. По заданному множеству строк $X = \{x_1, \ldots, x_n\}$ — представителей треков, построить $\mathcal{P}$-автомат, распознающий принадлежность произвольной строки y языку $[x_1] \cup \ldots \cup [x_n] = [X]$.

Решение задачи использует известный [11, Prop. 5.3] алгоритм построения минимального автомата $T_{\mathrm{st}}(X)$, допускающего язык строк $X = \{x_1, \ldots, x_n\}$. Очевидно, что автомат $T_{\mathrm{st}}(X) = (\mathrm{Pref}(X), \varepsilon, \{(p, a, pa)\}, X)$ допускает язык X. Здесь ε — начальное состояние, $a \in \Sigma$, $p, pa \in \mathrm{Pref}(X)$, где $\mathrm{Pref}(X) = \{u \in \Sigma^* \mid \exists\, i, v : i \in \{1, \ldots, n\}, v \in \Sigma^*, x_i = uv\}$ есть множество префиксов всех строк $x_i \in X$.

Языку треков $[X] = [x_1] \cup \ldots \cup [x_n]$ поставим в соответствие m языков строк — проекций на алфавиты Σ_i, $i=1,\ldots,m$, треков, входящих в $[X]$: $\Pi_i(X) = \{\pi_i(x_1), \ldots, \pi_i(x_n)\}$, $i=1,\ldots,m$.

Рассмотрим распределенный $\mathcal{P}$-автомат, построенный из автоматов $T_{\mathrm{st}}(\Pi_i(X))$. Такой $\mathcal{P}$-автомат не решает данной задачи, что видно из следующего примера.

Пример. Пусть $\Sigma = \{a, b, c\}$, $D = \{(a, b), (b, c)\}$, $\Sigma_1 = \{a, b\}$, $\Sigma_2 = \{c, b\}$, $X = \{abc, cba\}$. Тогда $\Pi_1(X) = \{ab, ba\}$, $\Pi_2(X) = \{bc, cb\}$. Распределенный $\mathcal{P}$-автомат с компонентами $T_{\mathrm{st}}(\Pi_1(X))$ и $T_{\mathrm{st}}(\Pi_2(X))$ допустит также треки $[acb], [bca] \notin [X]$. Следовательно, конечные языки треков, вообще говоря, нельзя распознать распределенными автоматами, и механизм отсева лишних треков обеспечивается суперпозицией распределенного $\mathcal{P}$-автомата с автоматом A_Φ.

Утверждение 4. Пусть автомат $T_{\mathrm{st}}(X)$ допускает язык строк $X = \{x_1, \ldots, x_n\}$. Тогда $\mathcal{P}$-автомат $T_{\mathrm{tr}}(X) = (T_{\mathrm{st}}(\Pi_1(X)), \ldots, T_{\mathrm{st}}(\Pi_m(X)), \Phi_{T_{\mathrm{tr}}})$ допускает язык треков $[X] = [x_1] \cup \ldots \cup [x_n] \subset M(\Sigma, D)$.

Здесь $\Pi_i(X) = \{\pi_i(x_1), \ldots, \pi(x_n)\}$, $i=1,\ldots,m$, — языки проекций строк из множества X на клики, и финальное множество имеет вид

$$\Phi_{T_{\mathrm{tr}}} = \{(q_1, \ldots, q_m) \in \Pi_1(X) \times \ldots \times \Pi_m(X) \mid \exists\, x \in X :$$
$$\varepsilon_i \xrightarrow{\pi_i(x)} q_i,\ i=1,\ldots,m\}. \qquad (1)$$

Автомат $T_{\text{tr}}(X)$ можно построить за время $O(m|X|\cdot\log d)$, где $|X|=\Sigma_{i=1}^{n}|x_i|$, d — максимальная степень состояний автоматов $T_{\text{st}}(\Pi_1(X)),\ldots,T_{\text{st}}(\Pi_m(X))$. Задержка автомата $Z(T_{\text{tr}}(X))=O(m\log d)$.

Доказательство. Из формулы (1) тривиально следует, что автомат $T_{\text{tr}}(X)$ допускает язык треков $[X]\subset M(\Sigma,D)$, так как состояние $(q_1,\ldots,q_m)\in\Phi_{T_{\text{tr}}}$ достигается тогда и только тогда, когда на вход автомата $T_{\text{tr}}(X)$ подается строка y, представляющая один из треков языка $[X]$.

Рассмотрим автомат TRIE_{tr}, строящий автомат $T_{\text{tr}}(X)$. В [11, Prop. 5.3] описан автомат TRIE_{st}, строящий такой минимальный автомат $T_{\text{st}}(X)$ для строк, который допускает язык строк $X=\{x_1,\ldots,x_n\}$ в алфавите Σ. Автомат TRIE_{st} по окончании чтения последовательности строк $X=\{x_1,\ldots,x_n\}$ попадает в финальное состояние $T_{\text{st}}(X)=(Q,\varepsilon,\delta,F)$, где Q — множество состояний, ε — начальное состояние, F — множество финальных состояний, δ — функция перехода автомата $T_{\text{st}}(X)$. Автомат TRIE_{st} строит автомат $T_{\text{st}}(X)$ за время $O(|X|\cdot\log d)$. Задержка автомата $Z(T_{\text{st}}(X))=O(\log d)$ и число состояний $\text{card}(Q)=O(|X|)$.

Для того чтобы в дальнейшем построить автомат для вычисления предиката $\Phi_{T_{\text{tr}}}$, модифицируем автомат TRIE_{st} так, чтобы вместо множества финальных состояний получалась структура — упорядоченный список финальных состояний $F^*=(x_1,\ldots,x_n)$, соответствующий чтению последовательности X. Обозначим этот модифицированный автомат $\text{TRIE}_{\text{st}}^{\wedge}$.

Положим $\text{TRIE}_{\text{tr}}=(\text{TRIE}_{\text{st}}^{\wedge},\ldots,\text{TRIE}_{\text{st}}^{\wedge},U)$ — распределенный $\mathcal{P}$-автомат. Автомат TRIE_{tr} при чтении последовательности строк $x_1,\ldots,x_n$ строит все локальные компоненты для требуемого автомата $T_{\text{tr}}(X)$, т.е. приходит в состояние $((Q_1,\varepsilon,\delta_1,F_1^*),\ldots,(Q_m,\varepsilon,\delta_m,F_m^*))$, где $(Q_1,\varepsilon,\delta_1,F_1^*)=T_{\text{st}}(\Pi_1(X)),\ldots$ $\ldots,(Q_m,\varepsilon,\delta_m,F_m^*)=T_{\text{st}}(\Pi_m(X))$ — искомые локальные компоненты и $F_i^*=(q_{i1},\ldots,q_{in})$, $i=1,\ldots,m$, — список финальных состояний автомата $T_{\text{st}}(\Pi_i(X))$. Каждый список финальных состояний соответствует порядку строк в $X=\{x_1,\ldots,x_n\}$, т.е. автомат $T_{\text{st}}(\Pi_i(X))$ попадает в состояние q_{ij} после прочтения строки x_j.

Тогда финальное множество $\mathcal{P}$-автомата $T_{\text{tr}}(X)$ есть $\Phi_{T_{\text{tr}}}=\{(q_{1j},\ldots$ $\ldots,q_{mj})\,|\,j=1,\ldots,n\}$, где $n=\text{card}(X)$, т.е. $\Phi_{T_{\text{tr}}}$ — множество наборов состояний частичных автоматов, входящих в списки финальных состояний автоматов $T_{\text{st}}(\Pi_1(X)),\ldots,T_{\text{st}}(\Pi_m(X))$ с одинаковыми номерами. Его можно легко построить по спискам $F_1^*,\ldots,F_m^*$, полученным $\mathcal{P}$-автоматом TRIE_{tr}, и предикат $(q_{1j},\ldots,q_{mj})\in\Phi_{T_{\text{tr}}}$ можно вычислить за время $O(m)$ с помощью автомата $A_{\Phi_{T_{\text{tr}}}}=T_{\text{st}}(\Phi_{T_{\text{tr}}})$, распознающего язык строк $\Phi_{T_{\text{tr}}}$.

Алгоритм построения $T_{\text{tr}}(X)$

Шаг 1. Применяется автомат TRIE_{tr} к последовательности X. За время $O(m\cdot|X|)$ будут построены автоматы $(U_1,\varepsilon,\delta_1,F_1^*)=T_{\text{st}}(\Pi_1(X)),\ldots$ $\ldots,(U_m,\varepsilon,\delta_m,F_m^*)=T_{\text{st}}(\Pi_m(X))$ со списками $F_1^*,\ldots,F_m^*$.

Шаг 2. По спискам $F_1^*,\ldots,F_m^*$ строится множество строк $\Phi_{T_{\text{tr}}}=$ $=\{q_{1j}\ldots q_{mj}\,|\,j=1,\ldots,n\}$, к которому применяется автомат TRIE_{tr}. Результатом является автомат $A_{\Phi_{T_{\text{tr}}}}=T_{\text{st}}(\Phi_{T_{\text{tr}}})$, распознающий финальные состояния автомата $T_{\text{tr}}(X)$ за время $O(m)$.

Следовательно, задержка $Z(T_{\text{tr}}(X)) = O(m \log d)$. Оценка времени построения $T_{\text{tr}}(X)$ вытекает из оценки времени построения $T_{\text{st}}(X)$.

ПОИСК СУФФИКСА ТРЕКА В СЛОВАРЕ

Задача 4. По словарю, т.е. множеству строк $X = \{x_1, \dots, x_n\}$ в алфавите Σ, построить $\mathcal{P}$-автомат, допускающий язык $[\Sigma^*]([x_1] \cup \ldots \cup [x_n])$.

Данное решение задачи также основано на известном [11] алгоритме решения задачи построения автомата для языка строк $\Sigma^* X$. Рассмотрим его подробнее. Как известно [10,11], автомат

$$D_{\text{st}}(X) = (\text{Pref}(X), \varepsilon, \text{Pref}(X) \cap \Sigma^* X, \{(p, a, h_X(pa)) | \, p \in \text{Pref}(X), \; a \in \Sigma\})$$

допускает язык $\Sigma^* X$. Здесь $h_X(v)$ — длиннейший суффикс строки v, принадлежащий $\text{Pref}(X) = U_{i=1}^{n} \text{Pref}(x_i)$.

Согласно [11, Th. 5.2] автомат $D_{\text{st}}(X)$ можно построить за время $O(|X| \cdot \log d)$, где d — максимальная степень состояний автомата $D_{\text{st}}(X)$. Задержка $Z(D_{\text{st}}(X)) = O(l \times \log d)$, где l — максимальная длина строк словаря X. Число состояний $\text{card}(Q) = O(|X|)$.

Известно, что граф переходов автомата $D_{\text{st}}(X)$ можно получить из графа переходов автомата $T_{\text{st}}(X) = (\text{Pref}(X), \varepsilon, \{(p, a, pa)\}, X)$, допускающего язык X, если к нему добавить ссылки и финальные вершины. Ссылка (q_1, q_2) — непомеченная дуга, где $q_1, q \in \text{Pref}(X)$, добавляется, если выполнено условие; q_2 — наибольший собственный суффикс строки q_1.

Множество финальных вершин автомата $D_{\text{st}}(X)$ наращивается, начиная с множества $F = X$ финальных вершин автомата $T_{\text{st}}(X) = (Q, \varepsilon, \delta, F)$, с помощью следующей итеративной процедуры дополнения. Если (q_1, q_2) — ссылка и состояние $q_2 \in \Sigma^* X$ — финальное, то состояние q_1 добавляется к множеству финальных вершин.

Ссылки образуют на множестве Q дерево ссылок $\Delta(X) = (Q, \delta')$ с корнем ε.

Обозначим $\Delta_X(q)$ множество состояний, достижимых из состояния q в дереве ссылок $\Delta(X)$.

Пусть числа $k_1, \dots, k_l$ $(1 \le k_i \le n = \text{card}(X), i = 1, \dots, l)$ таковы, что $x_{k_1} \in \text{Suff}\, x_{k_2}, \dots, x_{k_{l-1}} \in \text{Suff}\, x_{k_l}$ и для всякого $j \notin \{k_1, \dots, k_l\}$ имеет место $x_{k_l} \notin \text{Suff}\, x_j$. Легко видеть, что переход $\varepsilon \xrightarrow{y} x_{k_s}$ в автомате $D_{\text{st}}(X)$ для некоторого $1 \le s \le l$ имеет место тогда и только тогда, когда $y \in A^* x$ для всех $x \in \Delta_X(x_{k_s}) = \{x_{k_1}, \dots, x_{k_s}\}$ и $y \notin A^* x_j$ для $j \notin \{k_1, \dots, k_l\}$.

Модифицируем автомат $D_{\text{st}}(X)$, добавив к каждому его состоянию q информацию о финальных состояниях, принадлежащих множеству $\Delta_X(q)$: с каждым состоянием $q \in Q$ свяжем вектор $P_{D_{\text{st}}(X)}(q) = (p_{D_{\text{st}}(X)}(q, x_1), \dots, p_{D_{\text{st}}(X)}(q, x_n))$, где для $j = 1, \dots, n$

$$p_{D_{\text{st}}(X)}(q, x_j) = \begin{cases} 1, & \text{если } x_j \in \Delta_X(q), \\ 0 & \text{в противном случае.} \end{cases}$$

Векторы $P_{D_{\text{st}}(X)}(q)$ могут быть получены в процессе построения автомата $D_{\text{st}}(X)$. Каждое состояние q в $D_{\text{st}}(X)$ заменим парой $a_{D_{\text{st}}(X)}(q) = (q, P_{D_{\text{st}}(X)}(q))$, сохранив соответствующие переходы. Именно эти пары будем считать состояниями преобразованного автомата $D_{\text{st}}^{\wedge}(X) = (Q', \varepsilon, \delta', F)$. Множеством финальных состояний автомата $D_{\mathrm{st}}^{\wedge}(X)$ будет множество $F_{D_{\mathrm{st}}^{\wedge}(X)} = \{a_{D_{\mathrm{st}}(X)}(q) | q \in X\}$. Вычисляется автомат $D_{\mathrm{st}}^{\wedge}(X)$ за время $O(n|X| \cdot \log d)$.

Теперь рассмотрим заданное множество строк $X \subset \Sigma^*$ как множество представителей треков из словаря $[X] = \{[x_i] | i = 1, \ldots, n\} \subset M(\Sigma, D)$. Наша цель — построение $\mathcal{P}$-автомата $D_{\mathrm{tr}}(X)$, допускающего язык треков $[\Sigma^* X]$.

Утверждение 6. Пусть автомат $D_{\mathrm{st}}(X)$ допускает язык строк $X = \{x_1, \ldots, x_n\}$. Язык треков $[\Sigma^* X]$ распознается $\mathcal{P}$-автоматом $D_{\mathrm{tr}}(X) = (D_{\mathrm{st}}^{\wedge}(\Pi_1(X)), \ldots, D_{\mathrm{st}}^{\wedge}(\Pi_m(X)), \Phi_{D_{\mathrm{tr}}})$. Здесь $\Pi_i(X) = \{\pi_i(x_1), \ldots, \pi_i(x_n)\}$, $i = 1, \ldots, m$, — языки проекций на клики и финальное множество имеет вид

$$\Phi_{D_{\mathrm{tr}}} \left\{ (a_{D_{\mathrm{st}}}(\Pi_1(x)), \ldots, a_{D_{\mathrm{st}}}(\Pi_m(x))(q_m)) | \exists j : \prod_{i=1}^{m} p_{D_{\mathrm{st}}(\Pi_i(X))}(q_i, x_j) = 1 \right\}. \quad (2)$$

Для построения автомата необходимо время $O(mn|X| \cdot \log d)$, где $|X| = \Sigma_{i=1}^{n} |x_i|$, d — максимальная степень состояний автоматов $D_{\mathrm{st}}(\Pi_1(X)), \ldots$ $\ldots, D_{\mathrm{st}}(\Pi_m(X))$, l — максимальная длина строк в X. Задержка $Z(D_{\mathrm{tr}}(X)) = O(\max\{l \times \log d, mn\})$. Число состояний $O(|X|)$.

Доказательство. Приведем алгоритм построения автомата $D_{\mathrm{tr}}(X)$. Он сводится к построению автоматов $D_{\mathrm{st}}^{\wedge}(\Pi_1(X)), \ldots, D_{\mathrm{st}}^{\wedge}(\Pi_m(X))$ и построению автомата $A_{\Phi_{D_{\mathrm{tr}}}}$.

Шаг 1. В процессе чтения строк множества X строятся m автоматов $D_{\mathrm{st}}^{\wedge}(\Pi_i(X)) = (Q_i', \varepsilon_i, \delta_i, F_{D_{\mathrm{st}}^{\wedge}(\Pi_i(X))})$, $i = 1, \ldots, m$.

Шаг 2. Легко видеть, что глобальное состояние $(a_{D_{\mathrm{st}}(\Pi_1(X))}(q_1), \ldots$ $\ldots, a_{D_{\mathrm{st}}(\Pi_m(X))}(q_m))$ автомата $D_{\mathrm{tr}}(X)$ является финальным тогда и только тогда, когда существует число $1 \le j \le n$ такое, что для всех $i = 1, \ldots, m$ имеет место $p_{D_{\mathrm{st}}(\Pi_i(X))}(q_i, x_j) = 1$. Автомат $A_{\Phi_{D_{\mathrm{tr}}}}$, проверяющий это условие, можно построить за время $O(mn)$.

Автомат $D_{\mathrm{tr}}(X)$ построен за время $O(mn|X| \cdot \log d)$. Задержка $Z(D_{\mathrm{tr}}(X)) = O(\max\{l \times \log d, mn\})$.

Автомат над строками $D_{\mathrm{st}}(X)$ можно использовать для нахождения в произвольной строке y всех вхождений строк из языка X в качестве факторов строки y, так как в процессе чтения входной строки y автомат приходит в финальное состояние каждый раз, когда читается последняя буква вхождения какой-либо строки $x \in X$ в y. Однако автомат $D_{\mathrm{tr}}(X)$ нельзя использовать для нахождения в произвольном треке $[y]$ всех факторов — треков языка $[X]$. Его можно использовать для более частной задачи: для поиска в произвольной строке $y = y_1, \ldots y_{|y|} \in \Sigma^*$ всех префиксов $y = y_1, \ldots, y_k$ строки y, для которых существует $x_j \in [X]$ такое, что $[x_j] \in \mathrm{Suff}[y_1 \ldots y_k]$.

Утверждение 7. $\mathcal{P}$-автомат $D_{\mathrm{tr}}(X)$, построенный для словаря X, распознает в произвольной строке $y \in \Sigma^*$ все префиксы $y_1 y_2 \ldots y_k$ такие, что существует $x_j \in [X]$ такое, что $[x_j] \in \mathrm{Suff}[y_1 \ldots y_k]$. А именно $D_{\mathrm{tr}}(X)$ перерабатывает строку y в последовательность глобальных внутренних состояний $q_1, \ldots, q_{|y|}$ такую, что $q_k \in \Phi_{D_{\mathrm{tr}}}$, $k \le |y|$, тогда и только тогда, когда существует целое число $j \le n$ такое, что $[x_j] \in \mathrm{Suff}[y_1 \ldots y_k]$.

Заметим, что тем самым $D_{\text{tr}}(X)$ находит факторы (не все!) трека $[y]$, которые принадлежат $[X]$.

Представляет интерес задача, связанная с предыдущей: построить автомат, находящий все вхождения треков из словаря $[X]$ в качестве факторов в трек $[y]$, заданный своим представителем y. Однако для этого необходим просмотр всех префиксов трека $[y]$, число которых оценивается как $O(|y|^{\alpha})$, где α — число символов, входящих в наибольшую клику графа независимости алфавита Σ [7].

СПИСОК ЛИТЕРАТУРЫ

1. Dikert V., Rozenberg G. The book of traces // Handbook of Formal Languages. 3. Beyond Words. — N.Y.: Springer-Verlag — 1997. — 390 p.
2. Mazurkiewicz A. Concurrent program schemes and their interpretations // DAIMI Rep. PB. — Aarhus: Aarhus University, 1977. — **78**. — P. 16–23.
3. Zielonka W. Notes on finite asynchronous automata // RAIRO Inf. Th. Appl. — 1987. — **21**, N 2. — P. 99–135.
4. Duboc C. Mixed product and asynchronous automata // TCS. — 1986. — **48**. — P. 183–199.
5. Morin R. Decompositions of asynchronous systems // CONCUR, LNCS–1466, 1998. — P. 549–564.
6. Morin R. Concurrent automata vs. asynchronous systems // LNCS–3618, 2005. — P. 686–698.
7. Avellone A., Goldwurm M. Analysis of algorithms for the recognition of rational and context-free trace languages // Theoretical Inform. and Appl. — 1998. — **32M**. — P. 141–152.
8. Baeza-Yates R.A. Searching sequences // Theoretical Comput. Sci. — 1991. — **78**, N 2. — P. 363–376.
9. Crochemore M., Hancart C., Lecroq T. Algorithms on strings // Cambridge Univ. Press, 2007. — 58 p.
10. Crochemore M., Rytter W. Jewels of stringology // World Sci. Publ. Co. Rtc. Ltd., 2002. — 320 p.
11. Crochemore M., Hancart C. Automata for matching patterns. // Handbook of Formal Languages. 2. Linear Modeling. — N.Y.: Springer-Verlag, 1997. — P. 399–462.
12. Aho A.V., Corasick M.J. Efficient string matching: an aid to bibliographic research // Comm. of the ACM. — 1975. — **18**. — P. 333–340.
13. Shahbazyan K.V., Shoukourian Yu.H. Inclusion problems in trace monoids // CSIT. — 2009. — P. 65–69.

Поступила 07.07.2011