7

Лекция 23–24 марта

Регулярные выражения

Константин Чухарев

Дискретная математика — Весна 2027

Регулярные выражения

> *Аналитическая машина ткёт алгебраические узоры так же, как жаккардовый станок ткёт цветы и листья.*
>
> **— Ада Лавлейс**

Два описания языка

Автомат — *распознаватель*: читает слово и отвечает, принадлежит ли оно языку.

Нужна и *запись*: короткая формула, из которой язык восстанавливается без машины.

Стивен Клини ввёл регулярные выражения в 1951 году, чтобы не рисовать диаграммы: язык, заданный кружками и стрелками, хотелось сжимать до строки, которую можно продиктовать по телефону.

Автомат исполняет, выражение записывает.

Теорема Клини покажет, что это две стороны одного класса языков.

Регулярное выражение

Определение 1

Регулярное выражение над Σ строится по правилам:

- $\varnothing$ — пустой язык;
- ε — язык $\{\varepsilon\}$ из одного пустого слова;
- $a \in \Sigma$ — язык $\{a\}$;
- $R_1 \mid R_2$ — объединение языков;
- $R_1 R_2$ — конкатенация;
- R^* — звезда: ноль и более повторений.

Пример

Приоритет от сильной к слабой: $*$, конкатенация, $|$.

Скобки расставляют приоритет явно: $(a \mid b)c \neq a \mid (bc)$.

Примеры выражений

Пример

Над алфавитом $\{0, 1\}$:

- $(0 \mid 1)^*$ — все двоичные слова;
- 01^*0 — слова, начинающиеся и заканчивающиеся на 0;
- $(0 \mid 1)^*01$ — слова, заканчивающиеся на 01 — язык из детерминированной лекции;
- $(0 \mid 1)(0 \mid 1)^*$ — все непустые слова.

$\emptyset$, ε и 0^* — три разных языка: в первом нет слов, во втором слово длины 0, в третьем слова из нулей любой длины.

Теорема Клини

Теорема 1 (Клини)

Язык регулярен тогда и только тогда, когда он представим регулярным выражением: $\mathrm{AUT} = \mathrm{REG}$.

Доказательство

Доказательство — два построения, по одному на каждое включение.

От выражения к автомату. По выражению строится ε-НКА: базис для атомов, склейка по операциям.

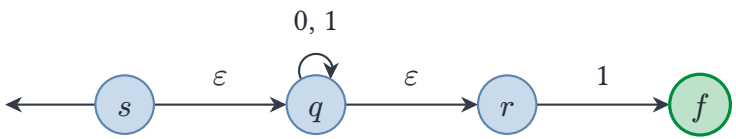
От автомата к выражению. Состояния превращаются в выражения, язык автомата собирается из них.

«Автоматный» и «регулярный» — синонимы: два описания одного класса.

От выражения к автомату

Конструкция Томпсона — структурная индукция по выражению.

- База: атомы a, ε, $\emptyset$ — автоматы из пары состояний.
- Объединение — новый старт с ε-переходами в оба автомата.
- Конкатенация — ε-переход из конца первого в старт второго.
- Звезда — возврат из конца в старт плюс обход пустого слова.



Пример

Автомат для выражения $(0 \mid 1)^*1$: цикл по любым символам и финальная единица.

Дальше — знакомый конвейер: ε-замыкание и конструкция подмножеств.

От автомата к выражению

Алгоритм Клини: состояния нумеруются, и через R_{ij}^{k} обозначаются слова, переводящие из q_i в q_j путями с промежуточными состояниями не выше k.

База — переходы автомата, шаг — рекуррентность:

$$R_{ij}^{k} = R_{ij}^{k-1} \cup R_{ik}^{k-1}\left(R_{kk}^{k-1}\right)^{*} R_{kj}^{k-1}.$$

Путь либо не заходит в q_k, либо проходит его цикл сколько угодно раз.

Это тот же приём, что в алгоритме Флойда—Уоршелла: динамическое программирование по промежуточным вершинам.

Разница в алгебре: вместо min и сложения — объединение, конкатенация и звезда.

Полное доказательство — на самостоятельное чтение: выражение выходит огромным, но для любого автомата существует.

Регулярные выражения вокруг нас

Пример

Идентификатор языка программирования: `[a-zA-Z_][a-zA-Z0-9_]*` — буква или подчёркивание, затем любые символы имени.

Тот же язык: буква (буква | цифра)*.

- Валидация форм: телефон, почтовый индекс, адрес почты.
- Поиск и замена в редакторах и утилитах командной строки.
- Подсветка синтаксиса и генераторы лексеров.

Всё это — теорема Клини в действии: запись компилируется в автомат, автомат исполняет распознавание.

Инженерия: от grep до лексера

Кен Томпсон скомпилировал регулярное выражение в автомат для редактора QED в 1968 году.

Тот же приём переехал в `ed`, затем в `grep`.

Лексер — первый компонент каждого компилятора: токены задаются регулярными выражениями, компилируются в ДКА и распознаются за $O(1)$ на символ.

Регулярные языки заканчиваются на лексере: вложенные скобки и грамматика языка требуют более сильных моделей.

Библиотеки с возвратом перебирают пути автомата по одному — на неудачных парах выражение–текст время взрывается экспоненциально.

В теории: выражение компилируется в ДКА, и распознавание линейно по тексту.

Нерегулярные языки

> *В одну и ту же реку нельзя войти дважды.*
>
> ***— Гераклит***

Конечная память имеет границу

Автомат не умеет считать неограниченно.

Пример

Язык $L = \{0^n 1^n \mid n \geq 0\}$: поровну нулей и единиц, все нули раньше.

Для распознавания нужно помнить разность «нули минус единицы» — а она ничем не ограничена.

«Не регулярен» — утверждение о всех автоматах сразу: ни один ДКА не распознаёт язык.

Для такого утверждения нужен инструмент — свойство, которым обладает *каждый* регулярный язык.

Цикл в диаграмме переходов

Пусть ДКА имеет p состояний и читает слово длины p.

Пройдено $p + 1$ позиций, а состояний p — по принципу Дирихле какое-то состояние повторилось.

Пример

Автомат с тремя состояниями читает 0001.

Пять позиций пути, три состояния: где-то автомат вернулся в то же место — и дальше идёт по той же колее, сколько бы раз ни прошёл цикл.

Повтор состояния — цикл, который можно пройти ноль, один или много раз.

Автомат разницы не заметит: конец пути тот же.

Лемма о накачке

Теорема 2 (Лемма о накачке)

Если язык L регулярен, то найдётся число p, такое что любое слово $w \in L$ длины $|w| \geq p$ разбивается как $w = xyz$, где:

- $|y| > 0$;
- $|xy| \leq p$;
- $xy^i z \in L$ для всех $i \geq 0$.

Доказательство

Докажем от принципа Дирихле.

Пусть ДКА для L имеет p состояний, на первых p символах слова w какое-то состояние s повторяется. Пусть x — префикс до первого вхождения s, y — цикл между двумя вхождениями, z — остаток. Тогда $|y| > 0$, $|xy| \leq p$, и накачка $xy^i z$ водит автомат по одному циклу — исход не меняется.

Накачка работает и на регулярных

Пример

Автомат чётности, слово 011, накачка $y = 11$.

Слова 011, 01111, 0 — все с чётным числом единиц, все в языке.

Лемма — свойство *каждого* регулярного языка: в нём найдётся накачиваемое разбиение каждого длинного слова.

Значит, если для языка разбиения нет ни при каком p — язык не регулярен.

Опровержение: язык $0^n 1^n$

Пример

Пусть $L = \{0^n 1^n\}$ регулярен с длиной накачки p.

Возьмём $w = 0^p 1^p \in L$. По лемме $w = xyz$ с $|xy| \le p$, $|y| > 0$.

1. Префикс xy целиком лежит в первых p символах — состоит только из нулей.
2. Значит, $y = 0^k$ для некоторого $k > 0$.
3. Накачанное слово $xy^2z = 0^{p+k}1^p$ должно лежать в L.
4. Но нулей в нём больше, чем единиц, — противоречие.

Следовательно, L не регулярен.

Схема опровержения: взять слово подлиннее p, накачать, получить слово вне языка.

Другие нерегулярные языки

- Палиндромы над $\{a, b\}$: слова a^n разводятся продолжением $ba^n - a^i ba^i$ палиндром, $a^j ba^i$ при $j \neq i$ нет.
- a^{2^n}: накачка $y = a^k$ даёт длины между 2^p и 2^{p+1}, а степеней двойки там нет.
- $a^n b^n c^n$: накачка в первых p символах портит блок a, а остальные блоки стоят на месте.

В каждом опровержении та же анатомия: длинное слово, накачка в префиксе, выход из языка.

Меняется только арифметика разрушения.

Границы леммы

Лемма о накачке даёт необходимое условие регулярности, но не достаточное.

Существуют нерегулярные языки, которым лемма удовлетворяет: на них опровержение не выходит.

Опровергнуть регулярность — можно, доказать — нет.

Конструктивный инструмент, работающий в обе стороны и строящий минимальный автомат, — теорема Майхилла–Нерода.

Её — следующая лекция.

Лекция 7: что мы поняли?

- Регулярное выражение — запись языка из атомов и трёх операций: $|$, конкатенация, звезда.
- Теорема Клини: автоматы и выражения описывают один класс, $\mathrm{AUT} = \mathrm{REG}$.
- От выражения к автомату — конструкция Томпсона, от автомата к выражению — алгоритм Клини, родня Флойда—Уоршелла.
- Лемма о накачке: длинное слово регулярного языка содержит накачиваемый цикл.
- $0^n 1^n$ не регулярен — первый язык, для которого конечная память доказанно не хватает.

Вопросы для самопроверки

1. Запишите регулярное выражение для слов над $\{0, 1\}$, содержащих подстроку 01.
2. Почему язык выражений $\varnothing$ и ε различны, а $\varnothing^*$ — это $\{\varepsilon\}$?
3. Какую роль в алгоритме Клини играет номер промежуточного состояния и что это за приём в терминах графов?
4. Накачкой докажите нерегулярность языка $a^n b^n c^n$.

Чтение к следующей паре

- m23 — «Регулярные выражения», «От выражения к автомату», «Лемма о накачке».
- Флеш-опросник — в начале лекции 8.
- Контрольная 2 (графы, автоматы и регулярные языки) — на следующей неделе, на отдельной паре.