Теория автоматов

Лекция 5: НКА, ДКА, регулярные языки

Дьулустан Никифоров

Кафедра ИТ
Северо-Восточный Федеральный Университет

Осень 2024

НКА = ДКА

Theorem

Если N — НКА (возможно, с ε-переходами), то существует ДКА, который распознает язык $L(N)$.

Proof:

Пусть $N = (Q, \Sigma, \delta, S, F), L(N) = A$.

Мы построим ДКА $M = (Q', \Sigma, \delta', S', F')$ такой, что $L(M) = A$.

НКА = ДКА

Theorem

Если N — НКА (возможно, с ε-переходами), то существует ДКА, который распознает язык $L(N)$.

Proof:

Пусть $N = (Q, \Sigma, \delta, S, F), L(N) = A$.

Мы построим ДКА $M = (Q', \Sigma, \delta', S', F')$ такой, что $L(M) = A$.

- Для $R \subseteq Q$ определим
  $E(R) =$
  $\{q \mid q$ можно достичь из R, пользуясь только ε-переходами$\}$;
- определим
  $\delta'(R, a) = \{q \in Q \mid q \in E(\delta(r, a))$ для некоторого $r \in R\}$;
- определим $S' = E(\{S\})$;

НКА = ДКА

Почему это доказательство работает?
Потому что на каждом шагу чтения строки, машина M всегда находится в состоянии, ровно соответствующем подмножеству состояний машины N в том же шагу.

НКА = ДКА

Практичный алгоритм, как перевести НКА N с ε-переходами в ДКА M:

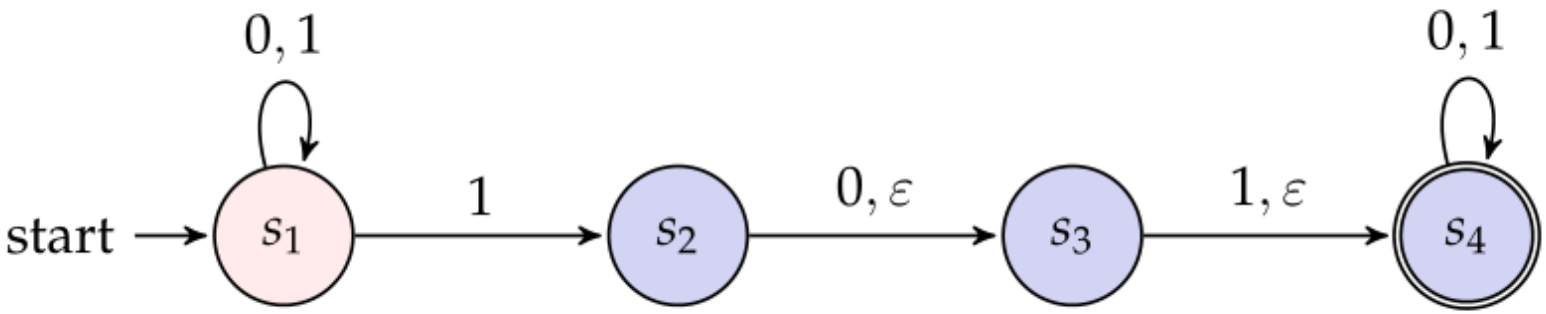
1. Создать начальное состояние (в M), состоящее из начального состояния из N плюс состояния, в которые можно из него попасть только по ε-переходам;
2. Добавить нехватающий переход в какое-нибудь из существующих состояний: посмотреть в какие состояния $(*)$ идет переход (в N) — сначала переход по букве, а потом можно делать переходы по ε-переходам $\rightarrow$ направить переход в то состояние M, которое представляет собой множество тех состояний $(*)$ из N (если такое состояние еще не существует, создать ее); может быть, что такого перехода из этого состояния вообще не существует в N — в этом случае надо делать переход в состояние, представляющее пустое множество состояний $(\emptyset)$

НКА = ДКА

3. Продолжать делать шаг 2, пока все созданные состояния не будут иметь все переходы.
4. Сделать принимающими те состояния (из M), которые содержат внутри хотя бы одно принимающее состояние из N.

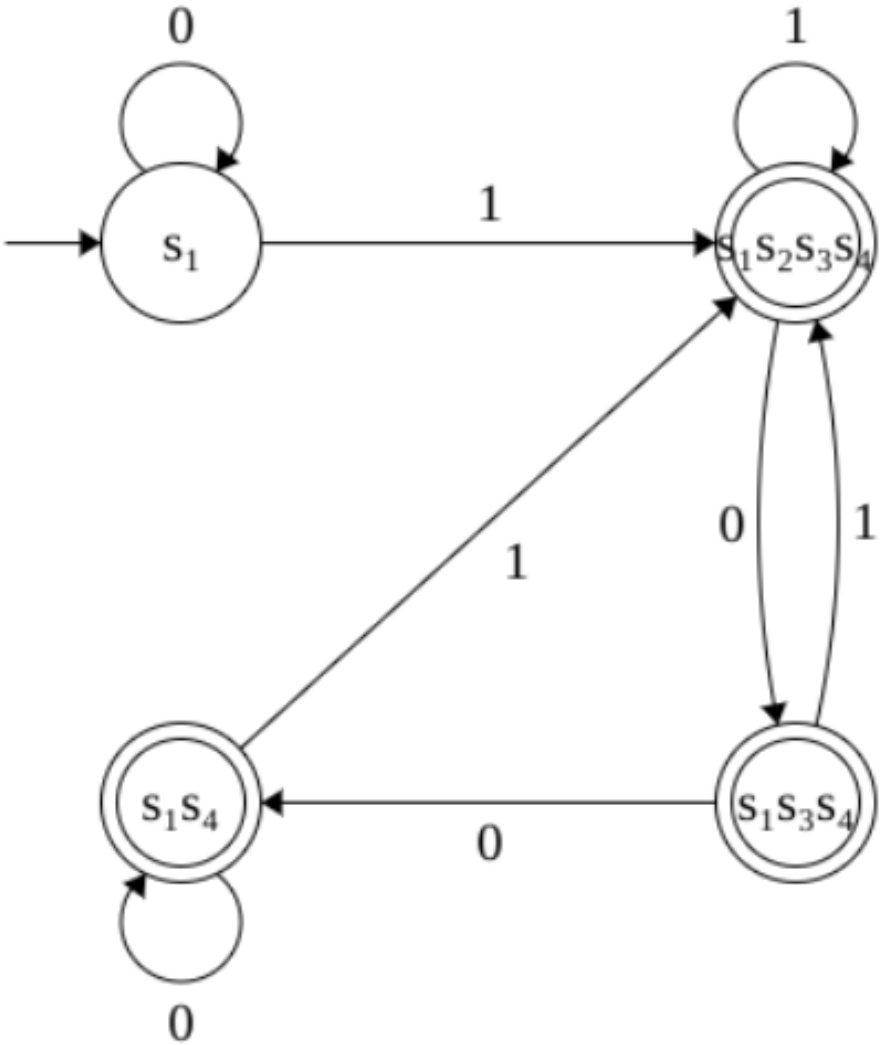
НКА = ДКА

На доске, применим этот метод для:



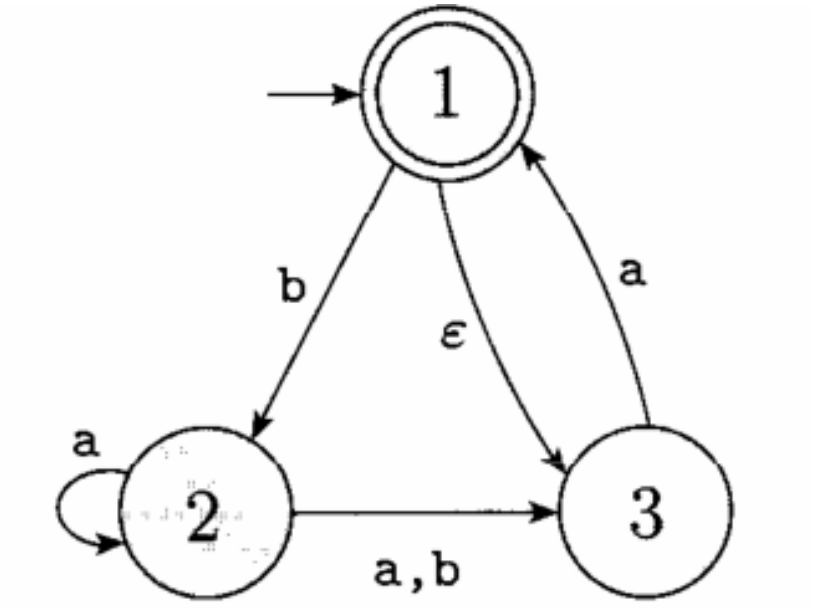
НКА = ДКА

На доске, применим этот метод для:



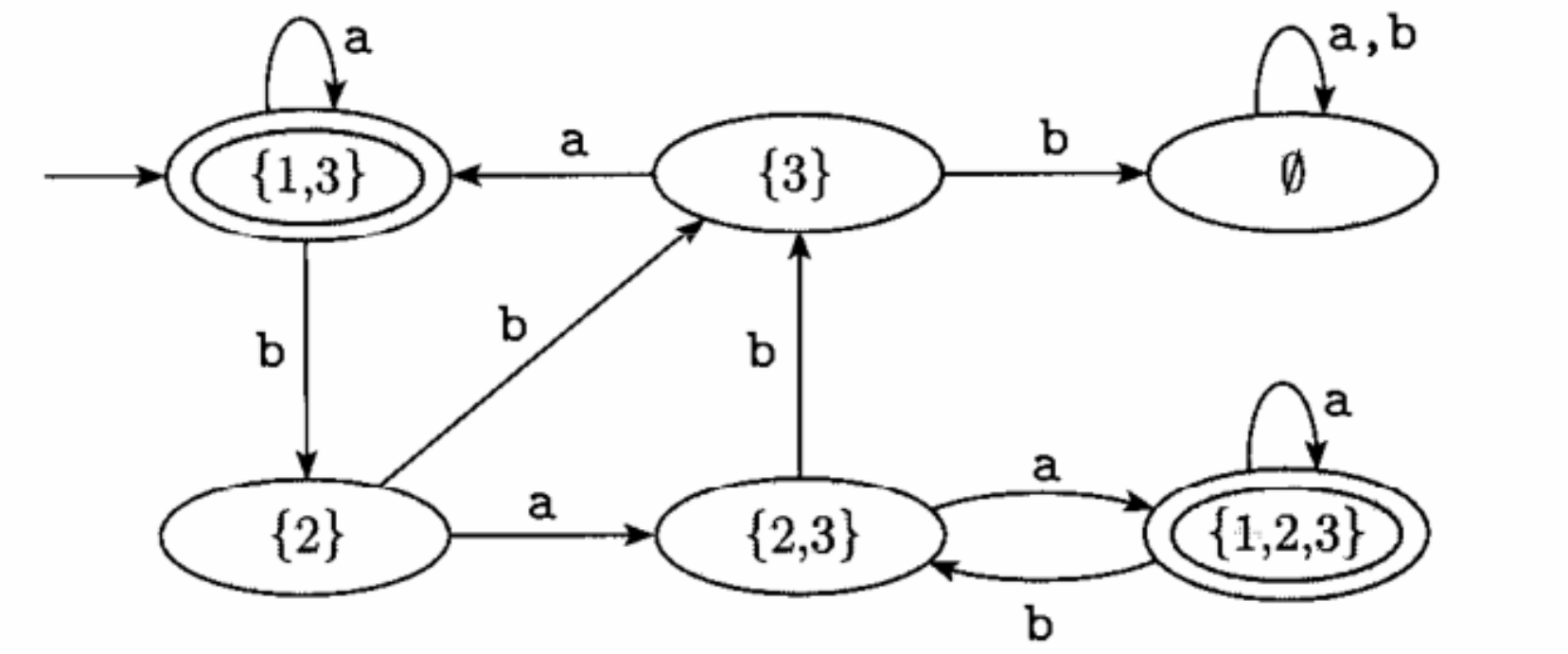
НКА = ДКА

Другой пример:



НКА = ДКА

Другой пример:



ДКА=НКА

- Мы доказали, что ДКА и НКА представляют одно и то же множество языков. Что это нам говорит про регулярные языки?

ДКА=НКА

- Мы доказали, что ДКА и НКА представляют одно и то же множество языков. Что это нам говорит про регулярные языки?
- *Язык регулярный тогда и только тогда, когда оно принимается каким-то НКА.*

  Language is regular if and only if it is accepted by some NFA.
- Главное, теперь мы получили новое мощное оружие для изучения свойств регулярных языков!

Theorem

Множество регулярных языков замкнуто под операцией конкатенации, т.е. A_1 и A_2 — регулярные языки $\Rightarrow A_1A_2$ — регулярный язык.

Proof:

Пусть $L(N_1) = A_1, L(N_2) = A_2, N_1 = (Q_1, \Sigma, \delta_1, S_1, F_1), N_2 = (Q_2, \Sigma, \delta_2, S_2, F_2)$. Построим $N = (Q, \Sigma, \delta, S, F)$ т.ч. $L(N) = A_1A_2$.

Theorem

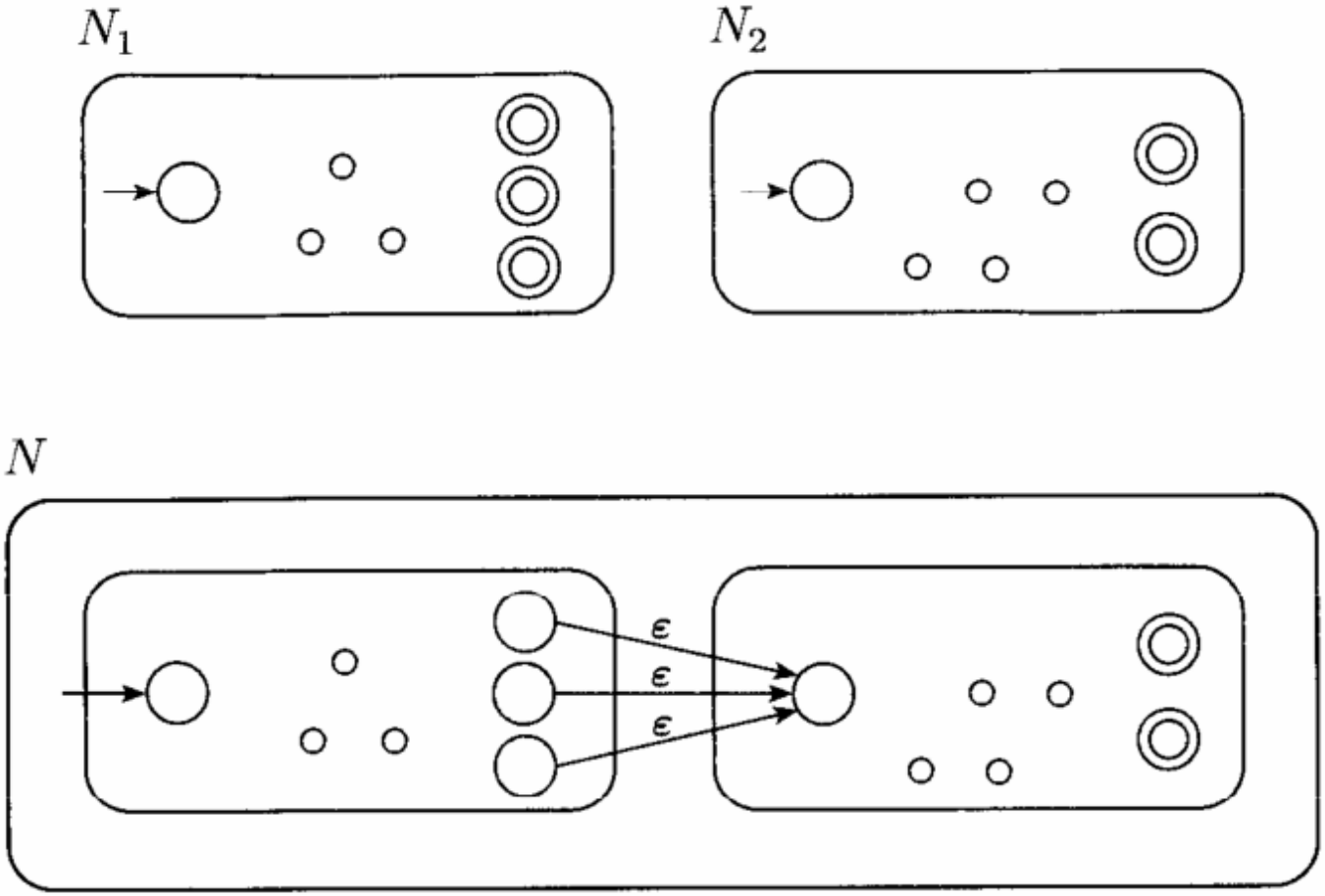
Множество регулярных языков замкнуто под операцией конкатенации, т.е. A_1 и A_2 — регулярные языки $\Rightarrow$ A_1A_2 — регулярный язык.

Proof:

Пусть $L(N_1) = A_1, L(N_2) = A_2, N_1 = (Q_1, \Sigma, \delta_1, S_1, F_1), N_2 = (Q_2, \Sigma, \delta_2, S_2, F_2)$. Построим $N = (Q, \Sigma, \delta, S, F)$ т.ч. $L(N) = A_1A_2$.

- $Q = Q_1 + Q_2$
- $S = S_1$
- $F = F_2$
- $\delta(q, a) = \delta_1(q, a)$ if $q \in Q_1$ and $q \notin F_1$,
  $\delta(q, a) = \delta_1(q, a)$ if $q \in F_1$ and $a \neq \varepsilon$,
  $\delta(q, a) = \delta_1(q, a) \cup \{S_2\}$ if $q \in F_1$ and $a = \varepsilon$,
  $\delta(q, a) = \delta_2(q, a)$ if $q \in Q_2$.

Регулярные языки



Theorem

Множество регулярных языков замкнуто под операцией замыкания, т.е. A_1 *— регулярный язык* $\Rightarrow$ A_1^* *— регулярный язык.*

Proof: Пусть $L(N_1) = A_1, N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$. Построим $N = (Q, \Sigma, \delta, q_0, F)$ т.ч. $L(N) = A_1^*$.

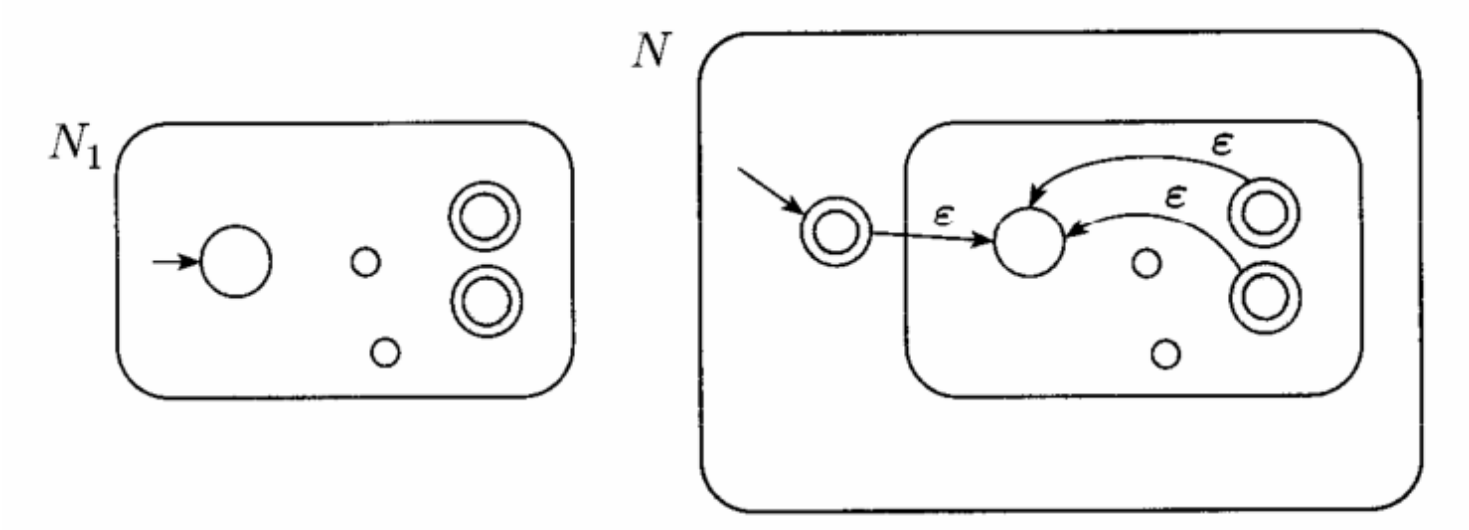
Theorem

Множество регулярных языков замкнуто под операцией замыкания, т.е. A_1 — регулярный язык $\Rightarrow$ A_1^* — регулярный язык.

Proof: Пусть $L(N_1) = A_1, N_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$. Построим $N = (Q, \Sigma, \delta, q_0, F)$ т.ч. $L(N) = A_1^*$.

- $Q = \{S\} \cup Q_1$,
- S — создаем новое состояние для этого,
- $F = \{S\} \cup F_1$,
- $\delta(q, a) = \delta_1(q, a)$ if $q \in Q_1$ and $q \notin F_1$,
  $\delta(q, a) = \delta_1(q, a)$ if $q \in F_1$ and $a \neq \varepsilon$,
  $\delta(q, a) = \delta_1(q, a) \cup \{S_1\}$ if $q \in F_1$ and $a = \varepsilon$,
  $\delta(q, a) = \{S_1\}$ if $q = S$ and $a = \varepsilon$,
  $\emptyset$ if $q = S$ and $a \neq \varepsilon$.

Регулярные языки



Регулярные языки

- Итак, мы уже очень много доказали о регулярных языках и конечных автоматах, которые распознают их:
  - Множество регулярных языков замкнуто под регулярными операциями: объединение, конкатенация, замыкание.
  - Каждый регулярный язык представим в виде некого ДКА, т.е. некий ДКА распознает его.
  - Каждый ДКА распознает регулярный язык.
  - Каждый НКА эквивалентен некому ДКА.
  - Регулярные языки = ДКА = НКА.
- Теперь мы добавим еще один важный элемент в это уравнение: *регулярные выражения*.

Регулярные операции

Вспомним определение регулярных операций с языками.
Пусть A и B языки. Тогда мы определяем такие **регулярные операции (regular operations)**:

- **Объединение (Union)** A и B:
  $A + B = \{x \mid x \in A \text{ или } x \in B\}$
- **Конкатенация (Concatenation)** A и B:
  $AB = \{xy \mid x \in A \text{ и } x \in B\}$
- **Замыкание/“звездочка” (Closure/Star)**:
  $A^* = \{x_1 x_2 \dots x_k \mid k \geq 0 \text{ и каждый } x_i \in A\}$.

Регулярные выражения

Регулярное выраженое — Regular Expression (RegEx). Я обычно просто называю их *регексами*.

Каждый Regex обозначает какой-то язык.

- $0 \to \{0\}$,
  $1 \to \{1\}$,
  $0 + 1 \to \{0\} + \{1\} = \{0, 1\}$,
  $0^* \to \{0\}^*$,
  $10^* \to \{1\}\{0\}^* = \{1, 10, 100, 1000, \dots\}$,
  $(0 + 1)0^* \to \{0 + 1\}\{0\}^* = ?$

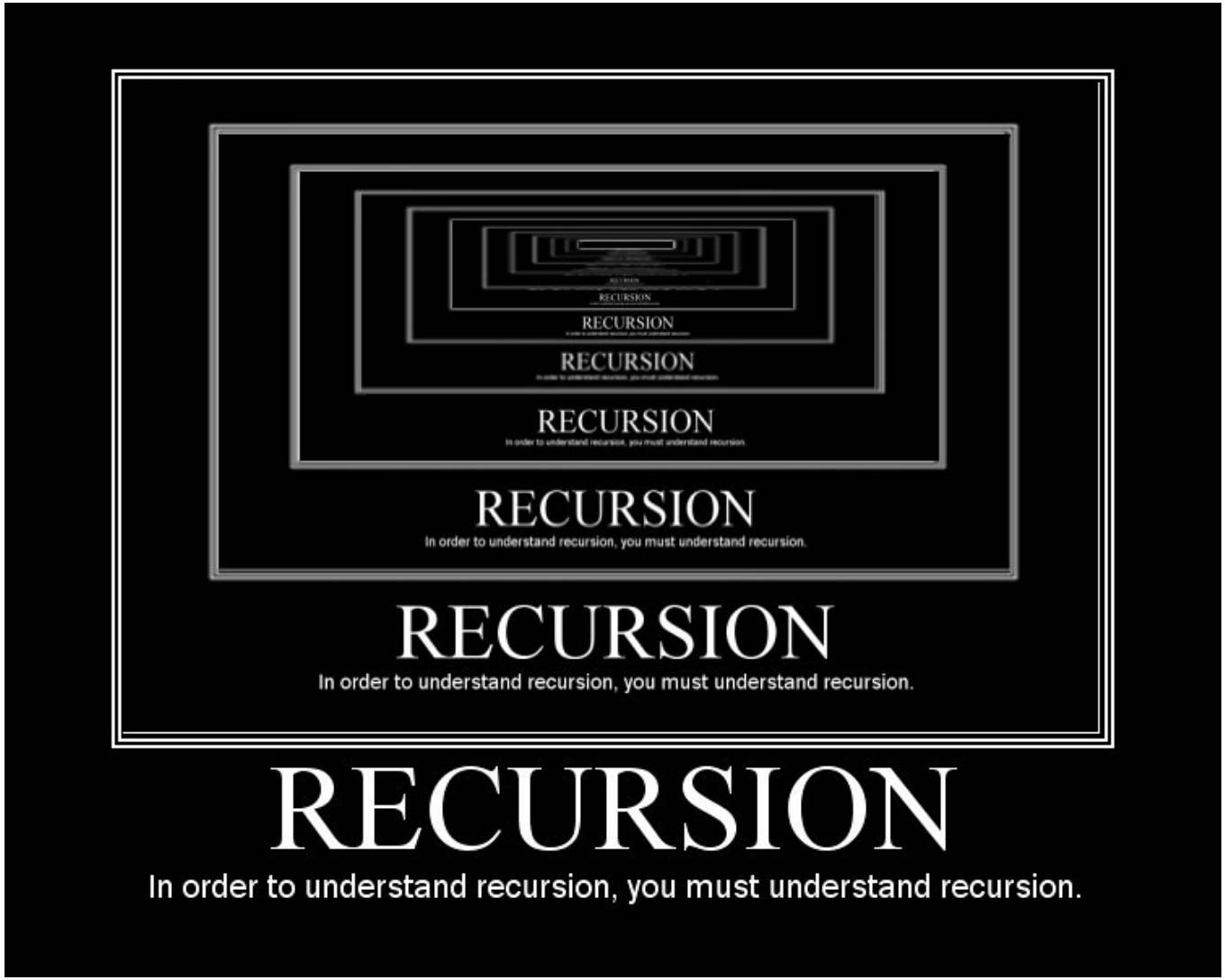
Регулярные выражения

Регулярное выраженое — Regular Expression (RegEx). Я обычно просто называю их *регексами*.

Каждый Regex обозначает какой-то язык.

- $0 \to \{0\}$,
  $1 \to \{1\}$,
  $0 + 1 \to \{0\} + \{1\} = \{0, 1\}$,
  $0^* \to \{0\}^*$,
  $10^* \to \{1\}\{0\}^* = \{1, 10, 100, 1000, \dots\}$,
  $(0 + 1)0^* \to \{0 + 1\}\{0\}^* = ?$
  $(0 + 1)0^* \to \{0 + 1\}\{0\}^* = \{0, 1, 00, 10, 000, 100, \dots\}$.

Регулярные выражения



Регулярные выражения

- Строгое определение регулярных выражений делается рекурсивно. Какие рекурсивные определения вы знаете?

Регулярные выражения

- Строгое определение регулярных выражений делается рекурсивно. Какие рекурсивные определения вы знаете?
- Есть алфавит Σ. Тогда мы говорим, что R — это **регулярное выражение (regular expression)**, если R является одним из следующих:
  - a для некоторого $a \in \Sigma$,
  - ε,
  - $\emptyset$,
  - $(R_1 + R_2)$, где R_1 и R_2 регулярные выражения,
  - $(R_1 R_2)$, где R_1 и R_2 регулярные выражения, или
  - (R_1^*), где R_1 регулярное выражение.

Регулярные выражения

- Строгое определение регулярных выражений делается рекурсивно. Какие рекурсивные определения вы знаете?
- Есть алфавит Σ. Тогда мы говорим, что R — это **регулярное выражение (regular expression)**, если R является одним из следующих:
  - a для некоторого $a \in \Sigma$,
  - ε,
  - $\emptyset$,
  - $(R_1 + R_2)$, где R_1 и R_2 регулярные выражения,
  - $(R_1 R_2)$, где R_1 и R_2 регулярные выражения, или
  - (R_1^*), где R_1 регулярное выражение.
- ε — соответствует языку, состоящему из одной пустой строки.
- $\emptyset$ — пустой язык.