Г-4. Автоматы

23 ноября 2024 г. 23:21

Помеченные орграфы

Помеченные орграфы

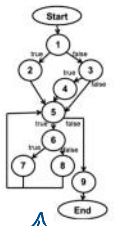
Появляются метки

- Ребра часто представляют взаимодействие между объектами-вершинами
- Когда моделируют несколько видов взаимодействия, ребра помечают, задавая функцию $\ell : E \to \Lambda$ из множества ребер в некоторое конечное множество меток
  - тройку $G = (V, E, \ell)$ называют помеченным (ор)графом
  - иногда говорят о раскрашивании ребер

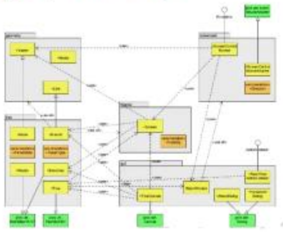
\# ℓ : сопоставляет ребру его вес

Примеры: слева control flow graph, справа UML class diagram

UML - universal mock-up language
Описание структур бизнес-процессов

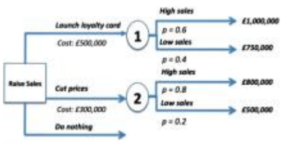


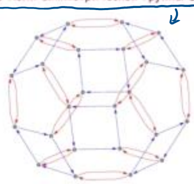
граф, который строит компилятор



Примеры: слева business decision tree, справа граф Кэли симметрической группы S_4

24 вершины = 4!





- Самая важная модель помеченного орграфа — конечный автомат

Детерминированный конечный автомат

Детерминированный конечный автомат

- Детерминированный конечный автомат (ДКА) — это пятерка $\mathcal{A} = (Q,\Sigma,\delta,s,T)$:
  - Q — непустое конечное множество состояний автомата
  - Σ — алфавит автомата (непустое конечное множество) → состоит из символов, которые как-то складываются в слова, которые обрабатывает автомат
  - $\delta : Q \times \Sigma \to Q$ — функция переходов
  - $s \in Q$ — начальное (стартовое) состояние
  - $T \subseteq Q$ — множество конечных (терминальных) состояний

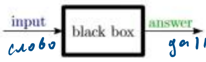
т.е. автомат переходит из некоторого состояния по некоторому символу в другое состояние.

- Q, Σ, δ задают помеченный орграф
  - Q — множество вершин
  - δ — множество помеченных ребер (метки — символы из Σ)
  - каждая вершина имеет степень исхода $|\Sigma|$ и ровно одно исходящее ребро с каждой меткой

т.е. конец ребра однозначно определен

Функционирование ДКА

- ДКА — простейший пример математической машины:



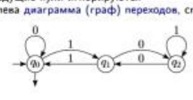
слово → дa/нет

  - входом для ДКА является произвольное слово $w \in \Sigma^*$
  - ДКА работает тактами (у него «дискретное время»)
  - перед первым тактом ДКА находится в начальном состоянии s
  - на i-ом такте ДКА обрабатывает символ $w[i]$:
    - переходит из текущего состояния q в состояние $\delta(q, w[i])$
    - иначе говоря, ДКА идет из вершины q по исходящему ребру с меткой $w[i]$
  - после обработки всего слова w автомат приходит в некоторое состояние t
  - ответом ДКА является булево значение $[t \in T]$ = {1; 0}
- машины, возвращающие булево значение, называются распознавателями
- Если ДКА на слове w возвращает
  - 1, то он читает / допускает w
  - 0, то он не читает / отвергает w
- Слова, которые читает ДКА $\mathcal{A}$, образуют множество $L(\mathcal{A}) \subseteq \Sigma^*$, которое называется языком, распознаваемым $\mathcal{A}$
  - (формальный) язык — это произвольное множество слов

Примеры ДКА

- ДКА, распознающий множество чисел, кратных 3, в двоичной записи:
  - ведущие нули игнорируются
  - слева диаграмма (граф) переходов, справа таблица переходов

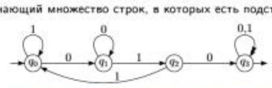
q_i - остаток от деления на 3



	0	1	T
q_0	q_0	q_1	1
q_1	q_2	q_0	0
q_2	q_1	q_2	0

! Достройте пример так, чтобы автомат не читал записи с ведущими нулями

- ДКА, распознающий множество строк, в которых есть подстрока 010:



! Перестройте пример так, чтобы распознавалось множество строк, в которых есть подпоследовательность 010

Комментарии и соглашения

- Так как ДКА — орграф, можно говорить о маршрутах
  - при этом вершины и ребра часто называют состояниями и переходами
  - каждый маршрут в $\mathcal{A}$ помечен словом $w \in \Sigma^*$, полученным конкатенацией меток составляющих маршрут ребер
- ★ Для каждого слова $w \in \Sigma^*$ существует единственный маршрут в ДКА $\mathcal{A} = (Q, \Sigma, \delta, s, T)$, помеченный w и начинающийся в s
  - ★ $\mathcal{A}$ читает $w \Leftrightarrow$ этот маршрут заканчивается в вершине из T
- Функцию переходов δ доопределяют на всём множестве $Q \times \Sigma^*$:
  - $\delta(q, a)$ — это конец ребра с меткой a, исходящего из q
  - ⇒ конец маршрута с меткой w и началом q обозначим за $\delta(q, w)$
  - часто пишут $q.w$ вместо $\delta(q, w)$, если автомат известен
  - например, $\mathcal{A}$ читает $w \Leftrightarrow s.w \in T$

$\delta(q, \theta) = q$

Вариации на тему ДКА.

Конечные автоматы с выходом

- Несложно сделать так, чтобы ДКА выдавал k вариантов ответа вместо 2:
  - множество T задает разбиение Q на два класса T и $Q \setminus T$, которым соответствуют ответы 1 и 0
  - вместо этого можно задать разбиение Q на k классов, которым соответствуют k возможных ответов
  - в предельном случае $k = |Q|$ ответ — это состояние (или его номер)
- Полученная такой модификацией ДКА машина — это конечный автомат с выходом
  - если автомат, проверяющий делимость на 3, будет возвращать номер текущего состояния, он будет вычислять функцию $x \bmod 3$
- Последовательность $\{a_i\}_1^\infty$ называется k-автоматной, если существует автомат с выходом, который по k-ичной записи числа n возвращает a_n (для любого n)
  - k-автоматные последовательности интересны тем, что о них можно доказывать теоремы автоматическим построением автоматов
  - → `https://github.com/hamousavi/Walnut`
  - вариант автомата с выходом, возвращающий свое состояние на каждом такте, называют машиной Мура
  - например, каждый элемент дисплея электронных часов управляется машиной Мура, совершающей один такт в секунду → пиксель горит или нет

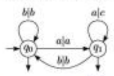
Конечные преобразователи

- Детерминированный конечный преобразователь

  он же детерминированный конечный трансдьюсер, машина Мили

  получается из ДКА добавлением выходного алфавита Γ и переопределением функции переходов (δ становится функцией из $Q \times \Sigma$ в $Q \times \Gamma$)
  - каждое ребро графа помечено парой букв (a, b), где $a \in \Sigma$, $b \in \Gamma$
  - по слову $w \in \Sigma^*$ преобразователь возвращает слово u длины $|w|$ в реальном времени
    - $u[i]$ — это буква, написанная на ребре, по которому автомат идет, читая $w[i]$
  - часто (но не обязательно) $\Gamma = \Sigma$

  Пример: преобразователь изменяет входное слово в алфавите $\{a, b\}$, заменяя в каждой последовательности букв a все буквы, кроме первой, на c



q_i - состояния

  - $aababaaa$ заменяется на $acbabacc$

… Об автоматах есть отдельный курс теории автоматов, а здесь мы обсудим только пару базовых для этой теории теорем

Недетерминированный конечный автомат

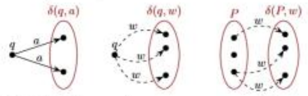
Недетерминированный конечный автомат

- ★ Что если отказаться от ограничения
  - из каждой вершины исходит ровно одно ребро с данной меткой?
- Если мы возьмем произвольный орграф, в котором выделены множество начальных вершин S и множество терминальных вершин T, а каждое ребро помечено буквой из алфавита Σ, то получится машина, называемая недетерминированным конечным автоматом (НКА)
  - НКА — это пятерка $\mathcal{A} = (Q,\Sigma,\delta,S,T)$, где $\delta \subseteq Q \times \Sigma \times Q$ — множество переходов
  - иногда δ удобно записывать как функцию $\delta : Q \times \Sigma \to 2^Q$
  - $\delta(q, a)$ — множество вершин, в которые из q ведет ребро с меткой a
    - $\delta(q, a)$ может быть пустым
  - доопределим функцию δ:
  - $\delta(q, w)$ — множество вершин, в которые из q ведет маршрут, помеченный w
  - $\delta(P, w)$, где $P \subseteq Q$, — множество вершин, в которые ведет маршрут, помеченный w и начинающийся в вершине из P

эл-т δ - помеченное ребро

$Q \times E \times Q$ ← конец

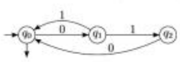
начало, помеченное ребро



- НКА $\mathcal{A} = (Q, \Sigma, \delta, S, T)$ читает/допускает слово $w \in \Sigma^*$, если существует (s, t)-маршрут с меткой w для некоторых $s \in S$, $t \in T$, то есть $\delta(S, w) \cap T \neq \varnothing$
- Язык $L(\mathcal{A})$ состоит из всех слов, читаемых $\mathcal{A}$

Пример НКА и комментарии

Данный НКА читает в точности те слова, которые можно разбить на блоки 01 и 010:



- ★ Недетерминированный выбор связан с состоянием q_1
  - Слово начинается с 1 или содержит 11 — автомат его не читает ($\delta(S, w) = \varnothing$)
  - $\delta(S, 010100) = \{q_1\}$ — автомат не читает 010100
  - $\delta(S, 010101) = \{q_0, q_2\}$ — автомат читает 010101
- ★ Термин «недетерминированный» применительно к алгоритму/машине означает, что вычисление может пойти различными путями
  - если очередной переход можно выбрать несколькими способами, НКА делает недетерминированный выбор т.е. вычисление может пойти различными путями
  - определение прочтения слова w означает, что НКА при выборе всегда «угадывает» так, чтобы в конце оказаться в терминальном состоянии
- ★ Определение чтения слова/распознавания языка при помощи НКА включает фундаментальную асимметрию между кванторами $\exists$ и $\forall$

- ★ Похоже, что НКА обладают бóльшими вычислительными возможностями, чем ДКА; тем не менее, это не так (см. следующий фрагмент)

Переход от НКА к ДКА

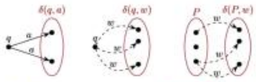
Теорема Рабина–Скотта

Теорема Рабина–Скотта

Для любого НКА существует ДКА, распознающий тот же самый язык.

- Доказательство:
  - возьмем произвольный НКА $\mathcal{B} = (Q, \Sigma, \delta, S, T)$
  - построим ДКА $\mathcal{A}$ такой, что $L(\mathcal{A}) = L(\mathcal{B})$
  - пусть $\mathcal{A} = (2^Q, \Sigma, \delta', S, T')$, где $\delta'(P, a) = \delta(P, a)$, $T' = \{P \in 2^Q \mid P \cap T \neq \varnothing\}$

напомним:



  - докажем, что $\delta'(P, w) = \delta(P, w)$ для любого слова w индукцией по $|w|$:
  - база индукции: для $|w| = 0$ имеем $\delta'(P, \lambda) = \delta(P, \lambda) = P$
  - шаг индукции: пусть $w = ua$, $a \in \Sigma$
    $\delta'(P, w) = \delta'(\delta'(P, u), a) = \delta'(\delta(P, u), a) =$
    $\delta(\delta(P, u), a) = \bigcup_{r \in \delta(P,u)} \delta(r, a) = \bigcup_{q \in P} \delta(q, ua) = \delta(P, ua)$
  - осталось заметить, что
    $L(\mathcal{A}) = \{w \in \Sigma^* \mid \delta'(S, w) \in T'\} = \{w \in \Sigma^* \mid \delta'(S, w) \cap T \neq \varnothing\} =$
    $\{w \in \Sigma^* \mid \delta(S, w) \cap T \neq \varnothing\} = L(\mathcal{B})$ □

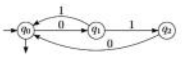
Достижимые состояния. Детерминирование НКА

- Пусть $\mathcal{A} = (Q, \Sigma, \delta, s, T)$ — ДКА
  - состояние $q \in Q$ достижимо, если существует $w \in \Sigma^*$ такое, что $q = s.w$
  - ⋆ т.е. если вершина q достижима из начальной вершины s
  - ⋆ недостижимые состояния можно удалить — это балласт, который занимает лишнее место и не влияет на функционирование автомата
  - ⋆ достижимые состояния находятся поиском из начальной вершины
- При построении ДКА $\mathcal{A}$, распознающего тот же язык, что и данный НКА $\mathcal{B}$, поиск совмещают с построением, получая $\mathcal{A}$ без недостижимых состояний:
  1. **для каждого** $P \subseteq Q$ $label(P) \leftarrow 0$
  2. $Q' \leftarrow \{S\}$
  3. **пока** $(\exists P \in Q' : label(P) = 0)$, **повторять**
  4. **для каждого** $a \in \Sigma$
  5. $\delta'(P, a) \leftarrow \bigcup_{q \in P} \delta(q, a)$
  6. $Q' \leftarrow Q' \cup \{\delta'(P, a)\}$
  7. $label(P) \leftarrow 1$
  8. $T' \leftarrow \{P \in Q' \mid P \cap T \neq \varnothing\}$

★ Обычно при использовании этого алгоритма Q' получается намного меньше, чем 2^Q; тем не менее, существуют НКА, для которых $Q' = 2^Q$

! Изучив доказательство теоремы Рабина–Скотта, придумайте, как вычислить ответ НКА $\mathcal{B}$ на слове w за время $O(|w| \cdot |Q|^2)$
  - это бывает выгоднее, чем построение и хранение большого ДКА

Пример



	0	1	T
q_0	q_1	$\varnothing$	1
q_1	$\varnothing$	q_0, q_2	0
$\varnothing$	$\varnothing$	$\varnothing$	0
q_0, q_2	q_0, q_1	$\varnothing$	1
q_0, q_1	q_1	q_0, q_2	1

Операции над языками

Моноид слов. Операции над языками

- Базовые определения:
  - Σ — конечное множество (алфавит)
  - Σ^* — множество всех конечных последовательностей элементов Σ (слов, строк)
  - $|w|$ — длина слова w, λ — пустое слово (длины 0)
  - на Σ^* задана операция конкатенации (умножения) слов
  - конкатенация ассоциативна, т.е. Σ^* — моноид относительно конкатенации
    - с единицей λ
    - Σ^* называют свободным моноидом или моноидом слов
  - $L \subseteq \Sigma^*$ — язык (конечный или бесконечный)
- Булевы операции над языками:
  - объединение $L_1 \cup L_2$
  - пересечение $L_1 \cap L_2$
  - дополнение $\bar{L} = \{w \in \Sigma^* \mid w \notin L\}$
  - разность $L_1 \setminus L_2 = L_1 \cap \bar{L}_2$
- Умножение языков:
  - $L_1 L_2 = \{uv \mid u \in L_1, v \in L_2\}$
    Пример: $\{ab, abc\}\{ba, cba\} = \{abba, abcba, abccba\}$
  - ⋆ конкатенация ассоциативна ⇒ умножение языков ассоциативно
    - и некоммутативно, разумеется
  - ⋆ естественно определяются степени языка: $L^n = L^{n-1}L$, $L^0 = \{\lambda\}$
- Итерация языка (Kleene star):
  - $L^* = \bigcup_{n=0}^{\infty} L^n$
  - ⋆ Σ^* — это действительно результат применения итерации к языку Σ
    Пример: $\{a, ab\}^* = a\Sigma^* \setminus \Sigma^* bb \Sigma^*$

Операции над языками (2)

★ Умножение языков дистрибутивно относительно объединения и пересечения:
  - $L_1(L_2 \cup L_3) = L_1L_2 \cup L_1L_3$, $(L_1 \cup L_2)L_3 = L_1L_3 \cup L_2L_3$
  - $L_1(L_2 \cap L_3) = L_1L_2 \cap L_1L_3$, $(L_1 \cap L_2)L_3 = L_1L_3 \cap L_2L_3$

⋆ Для итерации дистрибутивности нет, но верно следующее:
  - $(L_1 \cap L_2)^* \subseteq L_1^* \cap L_2^*$
  - $L_1^* \cup L_2^* \subseteq (L_1 \cup L_2)^*$
  - $L^{**} = L^*$ (идемпотентность)

- Другие полезные операции над языками:
  - $Lw^{-1} = \{u \mid uw \in L\}$ (правое деление на слово w)
  - $w^{-1}L = \{u \mid wu \in L\}$ (левое деление на слово w)
  - $\mathrm{pref}(L) = \{u \mid \exists w : uw \in L\}$ (префиксное замыкание)
  - $\mathrm{fact}(L) = \{u \mid \exists v, w : vuw \in L\}$ (подсловное замыкание)
  - $\mathrm{AD}(L) = \{u \notin L \mid$ все собственные подслова u принадлежат $L\}$ (антисловарь)
- Пусть $\preccurlyeq$ — отношение «быть подсловом» на Σ^*
  - ⋆ $\preccurlyeq$ — отношение порядка
  - ! докажите, что если $L = \mathrm{fact}(L)$, то язык $\mathrm{AD}(L)$ равен множеству минимальных элементов ЧУМа $(\bar{L}, \preccurlyeq)$
- Функция $\phi : \Sigma^* \to \Gamma^*$ — гомоморфизм, если $\phi(uv) = \phi(u)\phi(v)$ для любых $u, v \in \Sigma^*$
  - $\phi(L) = \{\phi(u) \mid u \in L\}$ — взятие гомоморфного образа

Регулярные языки. Теорема Клини.

Регулярные языки

- Пусть $\Sigma = \{a_1, \dots, a_n\}$
- Язык $L \subseteq \Sigma^*$ — регулярный, если он может быть получен применением конечного числа операций объединения, умножения и итерации к языкам $\varnothing, \{\lambda\}, \{a_1\}, \dots, \{a_n\}$
  - операции $\cup, \cdot, ^*$ также называются регулярными
- Можно взять замыкание любого множества языков $\mathbf{L} \subseteq 2^{\Sigma^*}$ относительно регулярных операций
- ★ Множество $\mathbf{R} \subset 2^{\Sigma^*}$ всех регулярных языков над Σ совпадает с замыканием множества всех конечных языков над Σ относительно регулярных операций
- Обычный способ записи регулярных языков — регулярные выражения:
  - символы $\varnothing, \lambda, a \in \Sigma$ являются регулярными выражениями
  - r, s — регулярные выражения $\Rightarrow (r)|(s)$, $(r) \cdot (s)$, $(r)^*$ — регулярные выражения
  - других регулярных выражений нет
    - $|$ — стандартный символ для перечисления альтернатив (соответствует операции $\cup$)
    - иногда вместо $|$ пишут $+$
- Для упрощения записи выражений договорились о приоритете операций:
  - * приоритетнее $\cdot$, $\cdot$ приоритетнее $|$
  - скобки, не меняющие порядок выполнения операций, опускаются
  - знак умножения также опускается
  - Пример: вместо $(((a) \mid (b)) \mid ((b) \cdot (c)))^*$ пишут $(a \mid b \mid bc)^*$

Теорема Клини

Теорема Клини

Язык регулярен тогда и только тогда, когда он распознается некоторым конечным автоматом.

Распознаваемость регулярных языков.

Регулярные языки распознаются автоматами

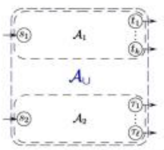
Докажем, что любой регулярный язык распознается конечным автоматом

- теорема Рабина–Скотта дает использовать ДКА и НКА вперемешку

План:

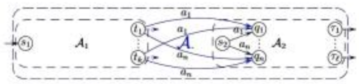
1. построить автоматы, распознающие языки $\varnothing, \{\lambda\}, \{a\}$
   ! постройте самостоятельно
2. по ДКА $\mathcal{A}_1 = (Q_1, \Sigma, \delta_1, s_1, T_1)$ и $\mathcal{A}_2 = (Q_2, \Sigma, \delta_2, s_2, T_2)$ построить автоматы, распознающие языки
   - $L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$
   - $L(\mathcal{A}_1) \cdot L(\mathcal{A}_2)$
   - $(L(\mathcal{A}_1))^*$

$\mathcal{A}_\cup = (Q_1 \cup Q_2, \Sigma, \delta_1 \cup \delta_2, \{s_1, s_2\}, T_1 \cup T_2)$
$L(\mathcal{A}_\cup) = L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$



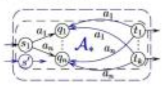
Регулярные языки распознаются автоматами (2)

$\mathcal{A}_\cdot = (Q_1 \cup Q_2, \Sigma, \delta, \{s_1\}, T_2)$ при $\lambda \notin L_2$,
$\mathcal{A}_\cdot = (Q_1 \cup Q_2, \Sigma, \delta, \{s_1\}, T_1 \cup T_2)$ при $\lambda \in L_2$,
где $\delta = \delta_1 \cup \delta_2 \cup \{(t, a, q) \mid t \in T_1, q \in Q_2, (s_2, a, q) \in \delta_2\}$
$L(\mathcal{A}_\cdot) = L(\mathcal{A}_1) \cdot L(\mathcal{A}_2)$



$\mathcal{A}_* = (Q_1 \cup \{s'\}, \Sigma, \delta', \{s_1, s'\}, T \cup \{s'\})$, где
$\delta' = \delta_1 \cup \{(t, a, q) \mid t \in T, q \in Q_1, (s_1, a, q) \in \delta\}$
- s' нужно только для распознавания λ

$L(\mathcal{A}_*) = (L(\mathcal{A}_1))^*$

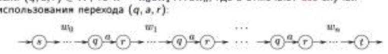


Регулярность автоматных языков

Регулярность автоматных языков

- Пусть $\mathcal{A} = (Q, \Sigma, \delta, s, T)$ — автомат; докажем, что $L(\mathcal{A}) \in \mathbf{R}$ индукцией по $|\delta|$
  База индукции: $|\delta| = 0$
  - $L(\mathcal{A}) = \{\lambda\} \in \mathbf{R}$ при $s \in T$ и $L(\mathcal{A}) = \varnothing \in \mathbf{R}$ при $s \notin T$

  Шаг индукции: $|\delta| = k$
  - по предположению индукции, языки, распознаваемые автоматами с менее чем k переходами (ребрами), регулярны
  - возьмем произвольный переход $(q, a, r) \in \delta$, пусть $\delta' = \delta \setminus \{(q, a, r)\}$; положим
    $\mathcal{A}_0 = (Q, \Sigma, \delta', s, T)$
    $\mathcal{A}_1 = (Q, \Sigma, \delta', s, \{q\})$
    $\mathcal{A}_2 = (Q, \Sigma, \delta', r, \{q\})$
    $\mathcal{A}_3 = (Q, \Sigma, \delta', r, T)$
  - языки $L(\mathcal{A}_0), L(\mathcal{A}_1), L(\mathcal{A}_2), L(\mathcal{A}_3)$ регулярны по предположению индукции
- Докажем, что $L(\mathcal{A}) = L(\mathcal{A}_0) \cup L(\mathcal{A}_1)a(L(\mathcal{A}_2)a)^*L(\mathcal{A}_3)$
  - пусть $w \in L(\mathcal{A})$ помечает (s, t)-маршрут W в $\mathcal{A}$, $t \in T$
  - если $(q, a, r) \notin W$, то $w \in L(\mathcal{A}_0)$
  - если $(q, a, r) \in W$, то $w = w_0 a w_1 \dots a w_n$, где a отмечают все случаи использования перехода (q, a, r):



$\Rightarrow$ $w_0 \in L(\mathcal{A}_1)$, $w_1, \dots, w_{n-1} \in L(\mathcal{A}_2)$, $w_n \in L(\mathcal{A}_3) \Rightarrow w \in L(\mathcal{A}_1)a(L(\mathcal{A}_2)a)^*L(\mathcal{A}_3)$
$\Rightarrow$ $w \in L(\mathcal{A}_0) \cup L(\mathcal{A}_1)a(L(\mathcal{A}_2)a)^*L(\mathcal{A}_3)$
  - $L(\mathcal{A}_0) \subseteq L(\mathcal{A})$ — очевидно
  - $w \in L(\mathcal{A}_1)a(L(\mathcal{A}_2)a)^*L(\mathcal{A}_3) \Rightarrow w = w_0 a w_1 \dots a w_n$ как на рисунке $\Rightarrow w \in L(\mathcal{A})$ □

Замкнутость относительно операций

Замкнутость R относительно операций

- Теорема Клини позволяет доказывать замкнутость **R** относительно операций
  - **R** замкнуто относительно дополнения:
  - если $\mathcal{A} = (Q, \Sigma, \delta, s, T)$ — ДКА, $L = L(\mathcal{A})$, то $\bar{L} = L(\bar{\mathcal{A}})$, где $\bar{\mathcal{A}} = (Q, \Sigma, \delta, s, Q \setminus T)$
  - **R** замкнуто относительно пересечения, потому что $L_1 \cap L_2 = \overline{(\bar{L}_1 \cup \bar{L}_2)}$
    - формулы де Моргана
  - **R** замкнуто относительно разности, потому что $L_1 \setminus L_2 = L_1 \cap \bar{L}_2$
  - ! докажите, у регулярного языка регулярными являются
    левые и правые частные
    гомоморфные образы
    префиксное и подсловное замыкание
    антисловарь
- ★ Объединение, пересечение и разность регулярных языков можно распознавать при помощи декартова произведения ДКА:
  - пусть $\mathcal{A}_1 = (Q_1, \Sigma, \delta_1, s_1, T_1)$, $\mathcal{A}_2 = (Q_2, \Sigma, \delta_2, s_2, T_2)$ — ДКА
  - $\mathcal{A}_\cup = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (s_1, s_2), Q_1 \times Q_2 \setminus (Q_1 \setminus T_1) \times (Q_2 \setminus T_2))$
  - $\mathcal{A}_\cap = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (s_1, s_2), T_1 \times T_2)$
  - $\mathcal{A}_\setminus = (Q_1 \times Q_2, \Sigma, \delta_1 \times \delta_2, (s_1, s_2), T_1 \times (Q_2 \setminus T_2))$