Модели вычислений

В.А. Захаров, Р.И. Подловченко

Лекция 1.

1. **Формальные языки. Операции над языками.**
2. **Разнообразие моделей вычислений.**
3. **Конечные автоматы. Автоматные языки.**
4. **Упрощение конечных автоматов.**
5. **Детерминированные конечные автоматы.**
6. **Минимизация детерминированных конечных автоматов.**

РАЗНООБРАЗИЕ МОДЕЛЕЙ ВЫЧИСЛЕНИЙ

машины Тьюринга

квантовые вычисления

молекулярные вычисления

мобильные амбиенты

алгебры процессов

автоматы Бюхи

π-исчисление

сети Петри

магазинные автоматы

временны́е автоматы

вероятностные автоматы

конечные автоматы

РАЗНООБРАЗИЕ МОДЕЛЕЙ ВЫЧИСЛЕНИЙ

машины Тьюринга
рекурсивно перечислимые языки
алгоритмически неразрешимые задачи

автоматы Бюхи
темпоральные логики
верификация моделей программ

магазинные автоматы
контекстно-свободные языки
синтаксический анализ языков

конечные автоматы
регулярные языки
поиск и распознавание

ФОРМАЛЬНЫЕ ЯЗЫКИ

Конечное непустое множество $\Sigma = \{a_1, \ldots, a_m\}$ будем называть алфавитом. Его элементы называются буквами.

Словом w в алфавите Σ называется любая конечная последовательность букв. Длина слова w обзначается записью $|w|$.

Слово, не содержащее ни одной буквы, называется пустым словом и обозначается символом ε.

Множество всех слов алфавита Σ обозначим записью Σ^*. Множество всех непустых слов алфавита Σ обозначим записью Σ^+.

Всякое множество слов L, $L \subseteq \Sigma^*$, будем называть формальным языком в алфавите Σ.

Формальные языки

Конкатенацией (сцеплением) слов u и v называется слово uv, полученное в результате приписывания слова v в конец слова u.

Слово x называется

- префиксом слова w, если $w = xy$;
- суффиксом слова w, если $w = yx$;
- подсловом слова w, если $w = yxz$,

где y и z — некоторые слова.
Запись x^n, где x — слово, а n — натуральное число, будет обозначать слово $\underbrace{xx\cdots x}_{n\ \text{раз}}$.

В частности, $x^0 = \varepsilon$.

Формальные языки

Примеры.

$\Sigma = \{a, b, c\}$ — алфавит.

$aabbcccbbaa$ — слово w в алфавите Σ^*.

w — конкатенация слов aab и $bcccbbaa$.

aab — префикс, $bbaa$ — суффикс, $bccc$ — подслово слова w.

$(aabc)^3 = aabcaabcaabc$

ФОРМАЛЬНЫЕ ЯЗЫКИ

К формальным языкам L_1 и L_2 применимы теоретико-множественные операции $L_1 \cap L_2$, $L_1 \cup L_2$, $L_1 \setminus L_2$, а также операция конкатенации

$$L_1 L_2 = \{uv \ : \ u \in L_1, \ v_{\in} L_2\},$$

и операции итерации

$$L_1^n = \{w^n \ : \ w \in L_1\}, \text{ где } n \geq 0;$$

$$L_1^* = \bigcup_{n=0}^{\infty} L_1^n.$$

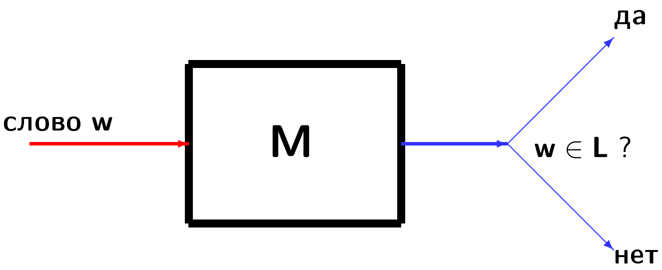
Также можно ввести одноместные операции

- $Pref(L) = \{u \ : \ w = uv, \ w \in L\}$;
- $Suff(L) = \{v \ : \ w = uv, \ w \in L\}$;
- $Sub(L) = \{x \ : \ w = uxv, \ w \in L\}$.

ОТКУДА БЕРУТСЯ ЯЗЫКИ?

И ЧТО НУЖНО УМЕТЬ УЗНАВАТЬ О ЯЗЫКАХ?

ОТКУДА БЕРУТСЯ ЯЗЫКИ?

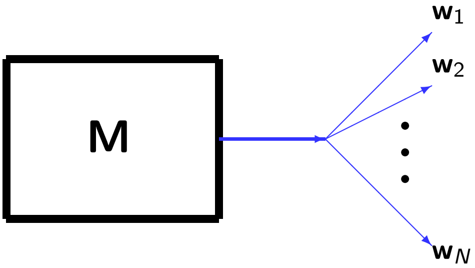


АВТОМАТ-РАСПОЗНАВАТЕЛЬ

ЯЗЫКА L

$$L = \{w \ : M(w) = \text{да}\}$$

ОТКУДА БЕРУТСЯ ЯЗЫКИ?

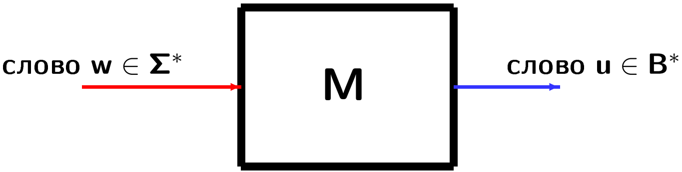


АВТОМАТ-ГЕНЕРАТОР

ЯЗЫКА L

$$L = \{w \ : w = output(M)\}$$

ОТКУДА БЕРУТСЯ ЯЗЫКИ?



АВТОМАТ-ПРЕОБРАЗОВАТЕЛЬ

$$L = \{u \ : u = M(w), \ w \in \Sigma^*\}$$

ОТКУДА БЕРУТСЯ ЯЗЫКИ?

Мы будем рассматривать модели вычислений, которые способны либо порождать формальные языки, либо распознавать принадлежность слов формальным языкам.

Если M — модель вычислений, то запись $L(M)$ обозначает язык, состоящий из всех слов w, которые порождаются моделью M или признаются моделью M допустимыми.

ЧТО НУЖНО УЗНАВАТЬ О ЯЗЫКАХ

Основные задачи анализа и синтеза моделей вычислений таковы

- для заданного языка L построить такую модель вычислений M, для которой верно $L = L(M)$ (проблема синтеза);
- для заданной модели вычислений M проверить $L(M) = \varnothing$ (проблема пустоты);
- для заданной модели вычислений M проверить $L(M) = \Sigma^*$ (проблема тотальности);
- для заданной пары моделей вычислений M_1, M_2 проверить $L(M_1) = L(M_2)$ (проблема эквивалентности);
- для заданной модели вычислений M и слова w проверить $w \in L(M)$ (проблема принадлежности).

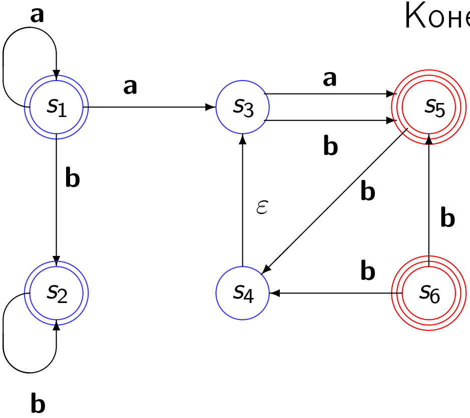
Конечные автоматы

Конечный автомат — это вычислительное устройство с конечной памятью.

Формально, конечный автомат — это система $\mathcal{A} = (\Sigma, S, I, F, T)$, где

- Σ — конечный алфавит;
- S — конечный множество состояний;
- I — множество начальных состояний, $I \subseteq S$;
- F — множество финальных состояний, $F \subseteq S$
- T — отношение переходов, $T \subseteq S \times (\Sigma \cup \{\varepsilon\}) \times S$;

КОНЕЧНЫЕ АВТОМАТЫ



Конечный автомат $\mathcal{A}$:

$$\Sigma = \{a, b\}$$
$$S = \{s_1, s_2, \ldots, s_6\}$$
$$I = \{s_1, s_2\}$$
$$F = \{s_5, s_6\}$$

Конечные автоматы

Для конечного автомата $\mathcal{A} = (\Sigma, S, I, F, T)$ условимся тройки (s', x, s'') из отношения переходов T называть переходами и изображать их записями вида $s' \xrightarrow{x} s''$.

Вычислением (прогоном, трассой) автомата $\mathcal{A}$ из состояния s_0 в состояние s_n называется любая конечная (в т.ч. пустая) последовательность переходов

$$run = s_0 \xrightarrow{x_1} s_1 \xrightarrow{x_2} \cdots \xrightarrow{x_n} s_n \ .$$

Будем говорить, что вычисление run прочитывает слово $w = x_1 x_2 \ldots x_n$, и условимся обозначать это вычисление записью $s_0 \xrightarrow{w}_* s_n$.

Конечные автоматы

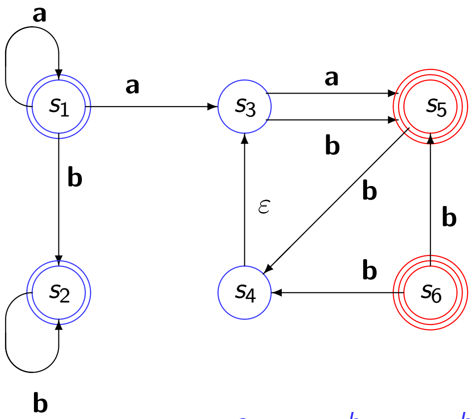
Вычисление $run = s_0 \xrightarrow{w}_* s_n$ конечного автомата $\mathcal{A} = (\Sigma, S, I, F, T)$ будем называть

- начальным, если $s_0 \in I$,
- финальным, если $s_n \in F$,
- успешным (или допускающим), если оно является и начальным, и финальным.

Язык автомата $\mathcal{A}$ — это множество слов

$$L(\mathcal{A}) = \{w \;:\; s_0 \xrightarrow{w}_* s_n \text{ — успешное вычисление } \mathcal{A}\}$$

КОНЕЧНЫЕ АВТОМАТЫ



$$run = s_1 \xrightarrow{a} s_3 \xrightarrow{b} s_5 \xrightarrow{b} s_4 \xrightarrow{\varepsilon} s_3 \xrightarrow{a} s_5$$

$$run = s_1 \xrightarrow{abba}_* s_5$$

КОНЕЧНЫЕ АВТОМАТЫ

Формальный язык L называется автоматным, если существует такой конечный автомат $\mathcal{A}$, для которого верно равенство $L = L(\mathcal{A})$.

Для автоматных языков некоторые задачи анализа и синтеза имеют эффективные решения.

Конечные автоматы

Теорема 1.1. Существует алгоритм, который для любого конечного автомата $\mathcal{A} = (\Sigma, S, I, F, T)$ и слова w проверяет принадлежность $w \in L(\mathcal{A})$ за время $O(|S| \cdot |w|)$.

Доказательство. Сводится к задаче проверки достижимости вершин в ориентированном графе.

Теорема 1.2. Существует алгоритм, который для любого конечного автомата $\mathcal{A} = (\Sigma, S, I, F, T)$ проверяет, верно ли, что $L(\mathcal{A}) = \varnothing$, за время $O(|S| \cdot |\Sigma|)$.

Доказательство. Сводится к задаче проверки достижимости вершин в ориентированном графе.

ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ

Конечные автоматы можно преобразовывать — упрощать, пополнять, минимизировать и пр. — путем эквивалентных преобразований.

Конечные автоматы $\mathcal{A}_1$ и $\mathcal{A}_1$ называются эквивалентными, если $L(\mathcal{A}_1) = L(\mathcal{A}_2)$.

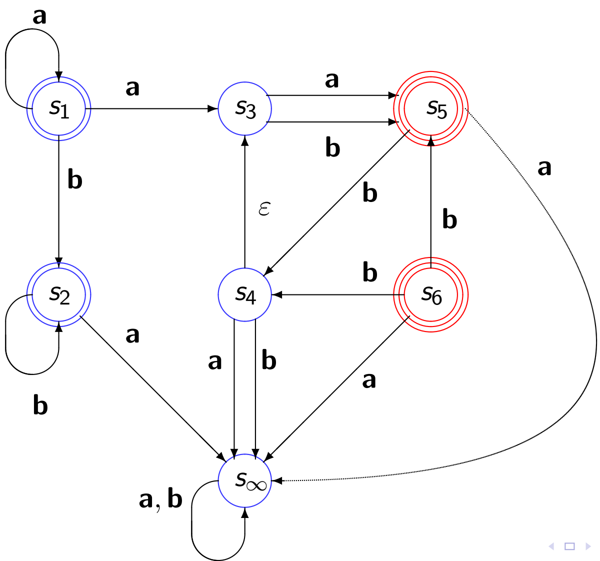
Конечный автомат $\mathcal{A} = (\Sigma, S, I, F, T)$ называется полным, если для любого состояния s, $s \in S$, и любой буквы a, $a \in \Sigma$, существует переход $s \xrightarrow{a} s' \in T$.

ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ

Утверждение 1.3. Каждый конечный автомат имеет эквивалентный ему полный автомат.

Доказательство. Введем новое **нефинальное** состояние s_∞ , и для каждого состояния $s, \ \ s \in S \cup \{s_{infty}\}$ и буквы $a, \ \ a \in \Sigma,$, добавим новый переход $(s \xrightarrow{a} s_\infty)$, если в автомате нет переходов вида $(s \xrightarrow{a} s')$. Очевидно, что в результате получим полный автомат, а множество успешных вычислений при этом не изменится.

ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ



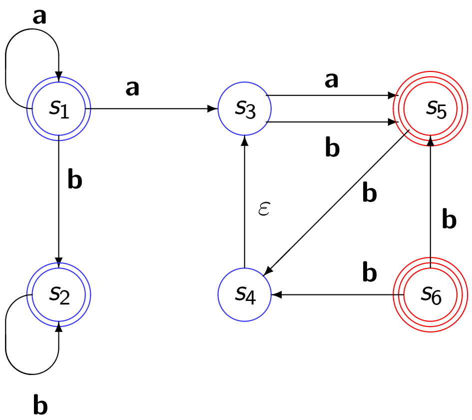
ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ

Состояние s конечного автомата $\mathcal{A}$ назовем несущественным, если через него не проходит ни одно успешное вычисление автомата. Конечный автомат, не имеющий несущественных состояний, будем называть приведенным.

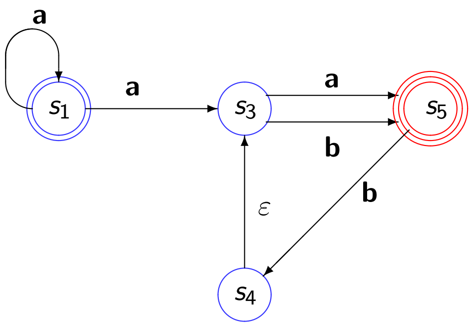
Утверждение 1.4. Каждый конечный автомат имеет эквивалентный ему приведенный автомат.

Доказательство. Удалим из S, I, F все несущественные состояния, а из отношения переходов T все переходы, содержащие несущественные состояния. Получим приведенный автомат с тем же множеством успешных вычислений.

ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ



ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ



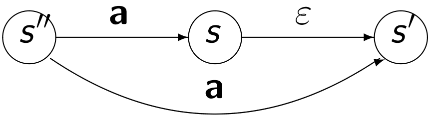
Приведенный автомат

ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ

Переходы вида $s \xrightarrow{\varepsilon} s'$ назовем ε-переходами.

Утверждение 1.5. Каждый конечный автомат имеет эквивалентный ему автомат без ε-переходов.

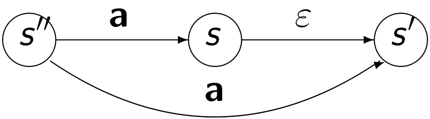
Доказательство. 1) Для каждой пары переходов $s \xrightarrow{\varepsilon} s'$ и $s'' \xrightarrow{a} s$ будем добавлять в автомат новый переход $s'' \xrightarrow{a} s'$.



Очевидно, что язык автомата при этом не изменяется.

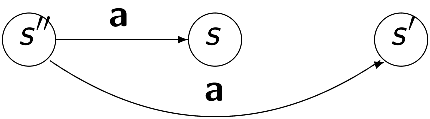
ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ

2) После того, как в автомате для каждой пары переходов $s \xrightarrow{\varepsilon} s'$ и $s'' \xrightarrow{a} s$ будет создан переход $s'' \xrightarrow{a} s'$, удалим все ε-переходы.



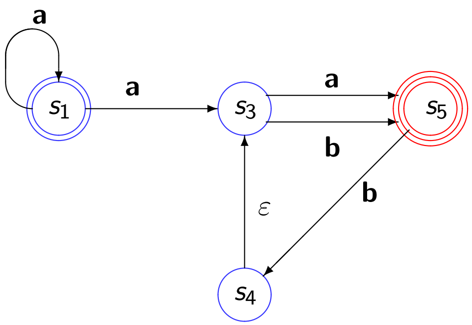
ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ

2) После того, как в автомате для каждой пары переходов $s \xrightarrow{\varepsilon} s'$ и $s'' \xrightarrow{a} s$ будет создан переход $s'' \xrightarrow{a} s'$, удалим все ε-переходы.

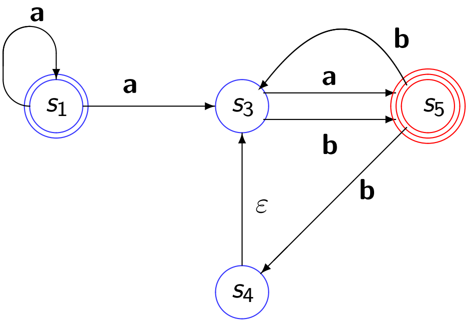


Очевидно, что язык автомата при этом не изменится. Вот и все.

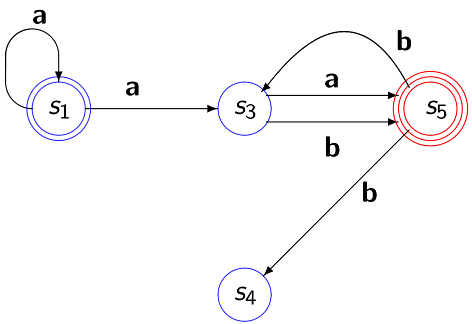
ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ



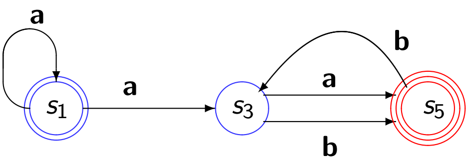
ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ



ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ



ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ



ПРЕОБРАЗОВАНИЯ КОНЕЧНЫХ АВТОМАТОВ

КАК УБЕДИТЬСЯ В ТОМ, ЧТО ДВА КОНЕЧНЫХ АВТОМАТА ЭКВИВАЛЕНТЫ?

И ДО КАКОЙ СТЕПЕНИ МОЖНО УПРОЩАТЬ КОНЕЧНЫЕ АВТОМАТЫ?

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Вначале получим ответы на эти вопросы для конечных автоматов специального вида.

Конечный автомат $\mathcal{A} = (\Sigma, S, I, F, T)$ называется детерминированным, если

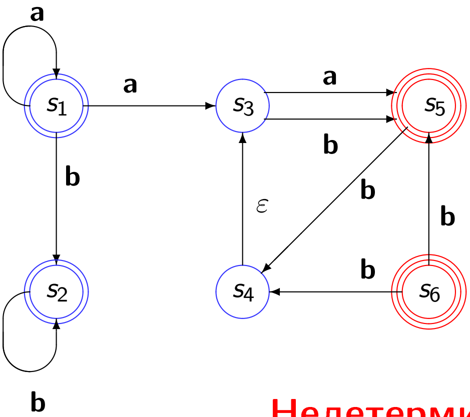
- $|I| \leq 1$, т.е. $\mathcal{A}$ имеет не более одного инициального состояния;
- $\mathcal{A}$ не имеет ε-переходов;
- для любого состояния $s, s \in S,$ и любой буквы a верно, что $|\{s' : (s, a, s') \in T\}| \leq 1$, т.е. отношение T является функцией $T : S \times \Sigma \to S$.

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Главная отличительная особенность детерминированных конечных автоматов:

для любого слова w и для любого состояния s автомат имеет **не более одного** вычисления $s \xrightarrow{w}_{*} s'$.

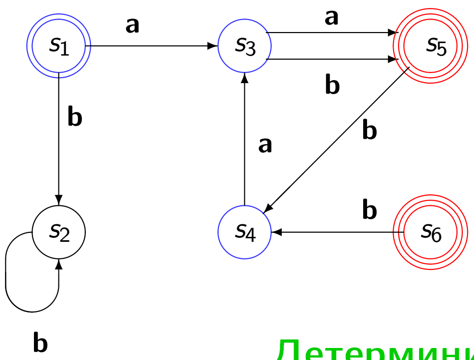
ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ



Недетерминированный

конечный автомат

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ



Детерминированный

конечный автомат

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Теорема 1.6. Существует алгоритм проверки включения языков $L(\mathcal{A}') \subseteq L(\mathcal{A}'')$ детерминированных конечных автоматов за время $O(n^2)$.

Доказательство. Пусть $\mathcal{A}' = (\Sigma, S', s_0', F', T')$ и $\mathcal{A}'' = (\Sigma, S'', s_0'', F'', T'')$ — пара **полных** детерминированных конечных автоматов, $|S'| \leq n$, $|S''| \leq n$.

Рассмотрим детерминированный конечный автомат $\mathcal{B} = (\Sigma, S, s_0, F, T)$, где

$S = S' \times S''$;

$s_0 = (s_0', s_0'')$;

$F = \{(s', s'') : s' \in F' \wedge s'' \notin F''\}$;

$T = \{(s_1', s_1'') \xrightarrow{a} (s_2', s_2'') : s_1' \xrightarrow{a} s_2' \in T' \wedge s_1'' \xrightarrow{a} s_2'' \in T''\}$.

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Индукцией по длине слова $w = a_1 a_2 \ldots a_k$ можно показать, что любой пары $s'_1 \in S'$, $s''_1 \in S''$ последовательность переходов

$$(s'_1, s''_1) \xrightarrow{a_1} (s'_2, s''_2) \xrightarrow{a_2} \cdots \xrightarrow{a_k} (s'_{k+1}, s''_{k+1})$$

является вычислением автомата $\mathcal{B}$

$$\Longleftrightarrow$$

последовательности переходов

$$s'_1 \xrightarrow{a_1} s'_2 \xrightarrow{a_2} \cdots \xrightarrow{a_k} s'_{k+1}$$
$$s''_1 \xrightarrow{a_1} s''_2 \xrightarrow{a_2} \cdots \xrightarrow{a_k} s''_{k+1}$$

являются вычислениями автоматов $\mathcal{A}'$ и $\mathcal{A}''$.

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Значит, $L(\mathcal{A}') \not\subseteq L(\mathcal{A}'') \iff$ существует такое слово $w = a_1 a_2 \ldots a_k$, для которого вычисление

$$s'_0 \xrightarrow{a_1} s'_1 \xrightarrow{a_2} \cdots \xrightarrow{a_k} s'_k, \;\; s'_k \in F'$$

автомата $\mathcal{A}'$ успешно, а вычисление

$$s''_0 \xrightarrow{a_1} s''_1 \xrightarrow{a_2} \cdots \xrightarrow{a_k} s''_k, \;\; s''_k \notin F''$$

автомата $\mathcal{A}''$ — нет $\iff$ автомат $\mathcal{B}$ имеет успешное вычисление

$$(s'_0, s''_0) \xrightarrow{a_1} (s'_0, s''_0) \xrightarrow{a_2} \cdots \xrightarrow{a_k} (s'_k, s''_k), \;\; (s'_k, s''_k) \in F$$

$\iff L(\mathcal{B}) \neq \varnothing$ (проверяется за время $O(n^2)$).

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Следствие. Существует алгоритм проверки эквивалентности детерминированных конечных автоматов за время $O(n^2)$.

Доказательство.
$L(\mathcal{A}_1) = L(\mathcal{A}_1) \;\Leftrightarrow\; L(\mathcal{A}_1) \subseteq L(\mathcal{A}_1) \wedge L(\mathcal{A}_2) \subseteq L(\mathcal{A}_1)$.

Эквивалентные детерминированные конечные автоматы обладают интересным свойством: их состояния попарно соответствуют друг другу.

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Теорема 1.7. Пусть $\mathcal{A}' = (\Sigma, S', s_0', F', T')$ и $\mathcal{A}'' = (\Sigma, S'', s_0'', F'', T'')$ — пара **приведенных** детерминированных конечных автоматов. Тогда для любого состояния s', $s' \in S'$ существует такое состояние s'', $s'' \in S''$, что автоматы $\mathcal{A}'[s'] = (\Sigma, S', s', F', T')$ и $\mathcal{A}''[s''] = (\Sigma, S'', s'', F'', T'')$ являются эквивалентными.

Доказательство. Пусть $s' \in S'$. Так как $\mathcal{A}'$ — приведенный автомат, существует такая пара слов u, v, что $s_0' \xrightarrow{u}_* s' \xrightarrow{v}_* s_n'$ — успешное вычисление автомата $\mathcal{A}'$.

Так как автоматы $\mathcal{A}'$ и $\mathcal{A}''$ эквивалентны, автомат $\mathcal{A}''$ также имеет успешное вычисление $s_0'' \xrightarrow{u}_* s'' \xrightarrow{v}_* s_n''$.

Покажем, что автоматы $\mathcal{A}'[s'] = (\Sigma, S', s', F', T')$ и $\mathcal{A}''[s''] = (\Sigma, S'', s'', F'', T'')$ эквивалентны.

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Допустим, что $w \in L(\mathcal{A}'[s'])$. Тогда существует успешное вычисление $s' \xrightarrow{w}_* s'_k$ автомата $\mathcal{A}'[s']$.

Тогда $s'_0 \xrightarrow{u}_* s' \xrightarrow{w}_* s'_k$ — успешное вычисление автомата $\mathcal{A}'$ (из определения s').

Тогда $s''_0 \xrightarrow{u}_* s'' \xrightarrow{w}_* s''_k$ — успешное вычисление автомата $\mathcal{A}''$ (из эквивалентности автоматов $\mathcal{A}'$ и $\mathcal{A}''$).

Тогда $s'' \xrightarrow{w}_* s''_k$ — успешное вычисление автомата $\mathcal{A}''[s'']$ (из определения s''), и, следовательно, $w \in L(\mathcal{A}''[s''])$.

Значит, $L(\mathcal{A}'[s']) \subseteq L(\mathcal{A}''[s''])$.

Аналогично показывается и включение $L(\mathcal{A}'[s']) \subseteq L(\mathcal{A}''[s''])$.
QED

МИНИМИЗАЦИЯ ДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ

А КАК МИНИМИЗИРОВАТЬ ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ

Детерминированный конечный автомат называется минимальным, если он имеет наименьшее число состояний среди всех эквивалентных ему детерминированных конечных автоматов.

МИНИМИЗАЦИЯ ДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ

КАК ПОСТРОИТЬ МИНИМАЛЬНЫЙ ДЕТЕРМИНИРОВАННЫЙ КОНЕЧНЫЙ АВТОМАТ?

Вначале избавимся от несущественных состояний и далее будем рассматривать только приведенные автоматы.

Состояния s', s'' детерминированного конечного автомата $\mathcal{A} = (\Sigma, S, s_0, F, T)$ будем называть эквивалентными, если эквивалентны автоматы $\mathcal{A}_{s'} = (\Sigma, S, s', F, T)$ и $\mathcal{A}_{s''} = (\Sigma, S, s'', F, T)$.

Минимизация детерминированных конечных автоматов

Будем использовать запись $s' \sim s''$ для обозначения эквивалентных состояний заданного автомата.

Эквивалентность состояний передается по наследству.

Утверждение 1.8. Если $s_1' \sim s_1''$ в приведенном детерминированном конечном автомате $\mathcal{A} = (\Sigma, S, s_0, F, T)$, и $s_1' \xrightarrow{a} s_2' \in T$, то существует такое состояние s_2'', что $s_1'' \xrightarrow{a} s_2'' \in T$ и $s_2' \sim s_2''$.

Доказательство. Самостоятельно

МИНИМИЗАЦИЯ ДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ

Эквивалентность состояний сохраняет свойство финальности.

Утверждение 1.9. Если $s_1' \sim s_1''$ в детерминированном конечном автомате $\mathcal{A} = (\Sigma, S, s_0, F, T)$, и $s' \in F$, то $s'' \in F$.

Доказательство. Самостоятельно

МИНИМИЗАЦИЯ ДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ

Эквивалентные состояния взаимозаменяемы.

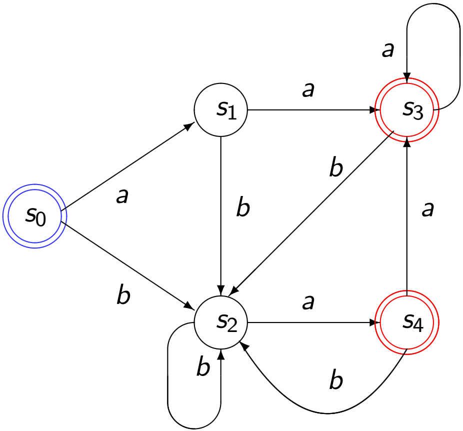
Утверждение 1.10. Если $s' \sim s''$ в детерминированном конечном автомате $\mathcal{A} = (\Sigma, S, s_0, F, T)$, и $s \xrightarrow{a} s' \in T$, то автомат $\widehat{\mathcal{A}} = (\Sigma, S, s_0, F, \widehat{T})$, где

$$\widehat{T} = (T \setminus \{s \xrightarrow{a} s'\}) \cup \{s \xrightarrow{a} s''\},$$

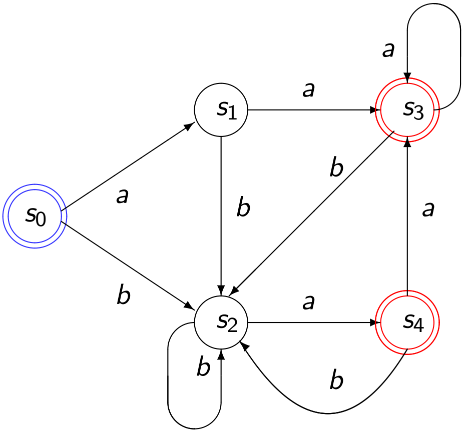
эквивалентен автомату $\mathcal{A}$.

Доказательство. Самостоятельно

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ



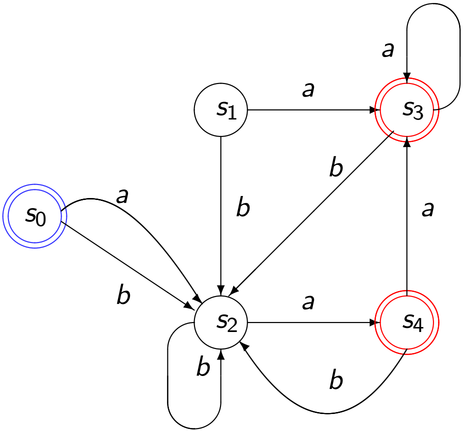
Детерминированные конечные автоматы



$s_1 \sim s_2$

$s_3 \sim s_4$

ДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ



$$s_1 \sim s_2$$

$$s_3 \sim s_4$$

Минимизация детерминированных конечных автоматов

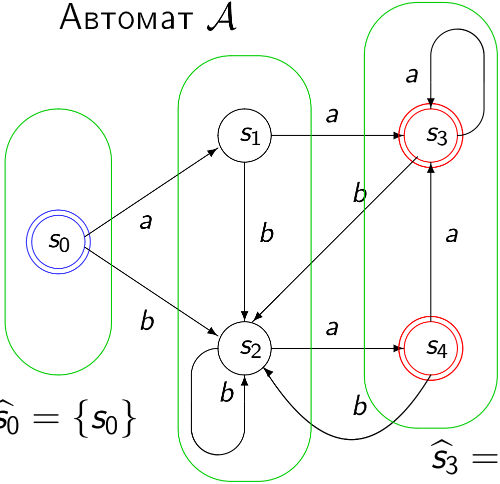
Для заданного детерминированного конечного автомата $\mathcal{A} = (\Sigma, S, s_0, F, T)$ построим автомат $\widehat{\mathcal{A}} = (\Sigma, \widehat{S}, \widehat{s}_0, \widehat{F}, \widehat{T})$ следующим образом:

- $\widehat{S} = S/\sim$ — фактор-множество S по отношению эквивалентности $\sim$;
- $\widehat{s}_0 = [s_0]_\sim$ — класс эквивалентности состояний, содержащий начальное состояние s_0;
- $\widehat{F} = F/\sim$ — множество классов эквивалентности, содержащих финальные состояния;
- $\widehat{T} = \{(\widehat{s}_1, x, \widehat{s}_2) : \exists s_1, s_2(\widehat{s}_1 = [s_1]_\sim \wedge \widehat{s}_2 = [s_2]_\sim \wedge (s_1, x, s_2) \in T)\}$

.

МИНИМИЗАЦИЯ ДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ

Автомат $\mathcal{A}$



Классы эквивалентности состояний

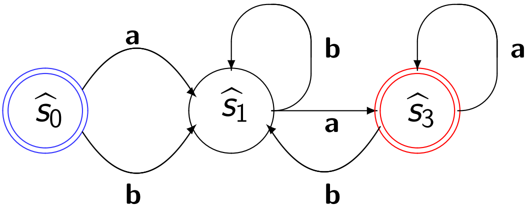
$\widehat{s}_0 = \{s_0\}$

$\widehat{s}_1 = \{s_1, s_2\}$

$\widehat{s}_3 = \{s_3, s_4\}$

Минимизация детерминированных конечных автоматов

Автомат $\widehat{\mathcal{A}}$



Минимизация детерминированных конечных автоматов

Теорема 1.11. $L(\mathcal{A}) = L(\widehat{\mathcal{A}})$.

Доказательство. Следует из предыдущих утверждений 1.8-1.10.

Вначале в каждом классе эквивалентности состояний автомата $\mathcal{A}$ выбирается представитель s, и все переходы, ведущие в состояния этого класса эквивалентности $[s]_\sim$, направляются в состояние s.

Затем удаляются несущественные состояния.

В результате получаем автомат $\widehat{\mathcal{A}}$.

МИНИМИЗАЦИЯ ДЕТЕРМИНИРОВАННЫХ КОНЕЧНЫХ АВТОМАТОВ

Теорема 1.12. $\widehat{\mathcal{A}}$ — минимальный детерминированный конечный автомат.

Доказательство. Предположим, что существует такой эквивалентный детерминированный конечный автомат $\mathcal{A}' = (\Sigma, S', s_0', F', T')$, что $|S'| < |\widehat{S}|$.

Тогда согласно теореме 1.7 о соответствии состояний эквивалентных автоматов для любого состояния $\widehat{s}$ автомата $\widehat{\mathcal{A}}$ существует такое состояние s' автомата $\mathcal{A}'$, что автоматы $\widehat{\mathcal{A}}[\widehat{s}]$ и $\mathcal{A}'[s']$ эквивалентны.

Но тогда с учетом $|S'| < |\widehat{S}|$ на основании принципа Дирихле приходим к выводу о существовании пары различных эквивалентных состоян ий автомата $\widehat{\mathcal{A}}$ вопреки тому, что по определению автомвтв $\widehat{\mathcal{A}}$ все его состояния попарно неэквивалентны.

Минимизация детерминированных конечных автоматов

Значит, такого эквивалентного автомата $\mathcal{A}' = (\Sigma, S', s_0', F', T')$, что $|S'| < |\widehat{S}|$, не существует.

Значит, $\widehat{\mathcal{A}}$ — минимальный детерминированный конечный автомат. QED

Следствие. Любые два эквивалентных минимальных детерминированных конечных автомата изоморфны.
Доказательство. Самостоятельно .

Т. е. каждый минимальный детерминированный конечный автомат для заданного автоматного языка определяется однозначно («с точностью до изоморфизма»).

Минимизация детерминированных конечных автоматов

Задачи проверки эквивалентности и минимизации конечных автоматов взаимносводимы.

Чтобы минимизировать конечный детерминированный автомат $\mathcal{A}$ достаточно

1. разбить его множество состояний на классы эквивалентности, применив алгоритм проверки эквивалентности детерминированных конечных автоматов,
2. построить минимальный фактор-автомат $\widehat{\mathcal{A}}$.

Чтобы проверить эквивалентность конечных детерминированных автоматов $\mathcal{A}_1$ и $\mathcal{A}_2$ достаточно

1. минимизировать оба автомата, применив алгоритм минимизации детерминированных конечных автоматов,
2. проверить изоморфность полученных минимальных автоматов.

ЗАДАЧИ

ЗАДАЧА 1

Докажите, что отношение эквивалентности состояний детерминированного конечного автомата $\mathcal{A} = (\Sigma, S, s_0, F, T)$ — это наибольшее по включению отношение эквивалентности R на множестве состояний S, которое удовлетворяет двум условиям для любых состояний s'_1, s'_2, s''_1, s''_2 и букв x:

1. $s'_1 R s''_1 \wedge s'_1 \in F \;\Rightarrow\; s''_1 \in F$,
2. $s'_1 R s''_1 \wedge s'_1 \xrightarrow{x} s'_2 \in T \wedge s''_1 \xrightarrow{x} s''_2 \in T \;\Rightarrow\; s'_2 R s''_2$,

ЗАДАЧА 2

Известно, что задачи проверки эквивалентности и минимизации для детерминированных конечных автоматов разрешимы за время $O(n \log n)$, где n — число состояний автоматов.

Предложите алгоритм минимизации детерминированных конечных автоматов, имеющий сложность $O(n^2)$. Обоснуйте его корректность и оценку сложности.

ЗАДАЧА 3

Известно, что для недетерминированных конечных автоматов теорема о единственности минимального автомата уже не имеет места.

Приведите пример двух минимальных, но неизоморфных недетерминированных конечных автоматов. Обоснуйте минимальность и эквивалентность предложенных автоматов.

КОНЕЦ ЛЕКЦИИ 1