

# Математические модели в лингвистике

## Математическая теория формальных языков

Мати Пентус, Александр Пиперски, Алексей Сорокин

МГУ им. М. В. Ломоносова, межфакультетский курс,  
осенний семестр 2015–2016 учебного года

# Мотивация

- Попробуем написать алгоритм автоматического образования множественного числа существительного в английском.
- Приблизительный алгоритм:
  - Если это исключение (*mouse, goose, ...*):  
Вернуть соответствующую форму (*mice, geese, ...*)

# Мотивация

- Попробуем написать алгоритм автоматического образования множественного числа существительного в английском.
- Приблизительный алгоритм:
  - Если это исключение (*mouse, goose, ...*):  
Вернуть соответствующую форму (*mice, geese, ...*)
  - Если слово кончается на шипящий (*-ch, -sh, -zh, -x, -s, -z*):  
Добавить *-es*

# Мотивация

- Попробуем написать алгоритм автоматического образования множественного числа существительного в английском.
- Приблизительный алгоритм:
  - Если это исключение (*mouse, goose, ...*):  
Вернуть соответствующую форму (*mice, geese, ...*)
  - Если слово кончается на шипящий (*-ch, -sh, -zh, -x, -s, -z*):  
Добавить *-es*
  - Если слово кончается на согласный + *y* (*ally, cry, ...*):  
Заменить финальный *-y* на *-ies*.

# Мотивация

- Попробуем написать алгоритм автоматического образования множественного числа существительного в английском.
- Приблизительный алгоритм:
  - Если это исключение (*mouse, goose, ...*):  
Вернуть соответствующую форму (*mice, geese, ...*)
  - Если слово кончается на шипящий (*-ch, -sh, -zh, -x, -s, -z*):  
Добавить *-es*
  - Если слово кончается на согласный + *y* (*ally, cry, ...*):  
Заменить финальный *-y* на *-ies*.
  - Иначе добавить к слову *-s*.
- Хочется задавать это формально.

# Мотивация

- Попробуем написать алгоритм автоматического образования множественного числа существительного в английском.
- Приблизительный алгоритм:
  - Если это исключение (*mouse, goose, ...*):  
Вернуть соответствующую форму (*mice, geese, ...*)
  - Если слово кончается на шипящий (*-ch, -sh, -zh, -x, -s, -z*):  
Добавить *-es*
  - Если слово кончается на согласный + *y* (*ally, cry, ...*):  
Заменить финальный *-y* на *-ies*.
  - Иначе добавить к слову *-s*.
- Хочется задавать это формально.
- То есть нужно математическое описание для условий вида:
  - Вставить *-a* в конце слова.
  - Заменить *-o* на *-ie* в первом слоге и т. д.

## Формальные определения

- $\Sigma$  — алфавит (произвольное конечное множество),  
его элементы — буквы.

## Формальные определения

- $\Sigma$  — алфавит (произвольное конечное множество), его элементы — буквы. Например:
  - $\Sigma = \{a\}$ , или  $\Sigma = \{a, b\}$  или  $\Sigma = \{a, b, c\}$ ;

## Формальные определения

- $\Sigma$  — алфавит (произвольное конечное множество), его элементы — буквы. Например:
  - $\Sigma = \{a\}$ , или  $\Sigma = \{a, b\}$  или  $\Sigma = \{a, b, c\}$ ;
  - $\Sigma$  — буквы кириллического или латинского алфавита;

## Формальные определения

- $\Sigma$  — алфавит (произвольное конечное множество), его элементы — буквы. Например:
  - $\Sigma = \{a\}$ , или  $\Sigma = \{a, b\}$  или  $\Sigma = \{a, b, c\}$ ;
  - $\Sigma$  — буквы кириллического или латинского алфавита;
  - $\Sigma$  — элементы международного фонетического алфавита IPA;

## Формальные определения

- $\Sigma$  — алфавит (произвольное конечное множество), его элементы — буквы. Например:
  - $\Sigma = \{a\}$ , или  $\Sigma = \{a, b\}$  или  $\Sigma = \{a, b, c\}$ ;
  - $\Sigma$  — буквы кириллического или латинского алфавита;
  - $\Sigma$  — элементы международного фонетического алфавита IPA;
  - $\Sigma$  — грамматические категории и т.д.

## Формальные определения

- $\Sigma$  — алфавит (произвольное конечное множество), его элементы — буквы. Например:
  - $\Sigma = \{a\}$ , или  $\Sigma = \{a, b\}$  или  $\Sigma = \{a, b, c\}$ ;
  - $\Sigma$  — буквы кириллического или латинского алфавита;
  - $\Sigma$  — элементы международного фонетического алфавита IPA;
  - $\Sigma$  — грамматические категории и т.д.
- Слова — конечные последовательности элементов  $\Sigma$ ,  $\Sigma^*$  — множество всех слов.  
 $\varepsilon$  — пустое слово (слово длины 0).

## Формальные определения

- $\Sigma$  — алфавит (произвольное конечное множество), его элементы — буквы. Например:
  - $\Sigma = \{a\}$ , или  $\Sigma = \{a, b\}$  или  $\Sigma = \{a, b, c\}$ ;
  - $\Sigma$  — буквы кириллического или латинского алфавита;
  - $\Sigma$  — элементы международного фонетического алфавита IPA;
  - $\Sigma$  — грамматические категории и т.д.
- Слова — конечные последовательности элементов  $\Sigma$ ,  $\Sigma^*$  — множество всех слов.  
 $\varepsilon$  — пустое слово (слово длины 0).
- Языки — произвольные подмножества  $\Sigma^*$ .

# Формальные определения

- $\Sigma$  — алфавит (произвольное конечное множество), его элементы — буквы. Например:
  - $\Sigma = \{a\}$ , или  $\Sigma = \{a, b\}$  или  $\Sigma = \{a, b, c\}$ ;
  - $\Sigma$  — буквы кириллического или латинского алфавита;
  - $\Sigma$  — элементы международного фонетического алфавита IPA;
  - $\Sigma$  — грамматические категории и т.д.
- Слова — конечные последовательности элементов  $\Sigma$ ,  $\Sigma^*$  — множество всех слов.  
 $\varepsilon$  — пустое слово (слово длины 0).
- Языки — произвольные подмножества  $\Sigma^*$ .
- Нам также понадобятся функции из  $\Sigma^*$  в  $\Sigma^*$  и бинарные отношения на множестве  $\Sigma^*$ , то есть подмножества  $\Sigma^* \times \Sigma^*$ .

# Регулярные выражения

Пусть зафиксирован конечный алфавит  $\Sigma$ .

- Базовые регулярные выражения: элементы алфавита,
- Также есть константы 0 (пустой язык) и 1 (язык, содержащий только пустое слово  $\varepsilon$ ).

# Регулярные выражения

Пусть зафиксирован конечный алфавит  $\Sigma$ .

- Базовые регулярные выражения: элементы алфавита,
- Также есть константы 0 (пустой язык) и 1 (язык, содержащий только пустое слово  $\varepsilon$ ).
- Бинарные операции:  $|$  (объединение) и  $\cdot$  (конкатенация или приписывание):  $u \cdot v = uv$ ,

# Регулярные выражения

Пусть зафиксирован конечный алфавит  $\Sigma$ .

- Базовые регулярные выражения: элементы алфавита,
- Также есть константы 0 (пустой язык) и 1 (язык, содержащий только пустое слово  $\varepsilon$ ).
- Бинарные операции:  $|$  (объединение) и  $\cdot$  (конкатенация или приписывание):  $u \cdot v = uv$ ,
- Унарная операция  $*$  (итерация, взять любое количество раз):  $L^*$  состоит из слов вида  $u_1 \dots u_r$ , где  $r \in \mathbb{N}$ ,  $u_1, \dots, u_r \in L$ .

# Регулярные выражения

Пусть зафиксирован конечный алфавит  $\Sigma$ .

- Базовые регулярные выражения: элементы алфавита,
- Также есть константы 0 (пустой язык) и 1 (язык, содержащий только пустое слово  $\varepsilon$ ).
- Бинарные операции:  $|$  (объединение) и  $\cdot$  (конкатенация или приписывание):  $u \cdot v = uv$ ,
- Унарная операция  $*$  (итерация, взять любое количество раз):  $L^*$  состоит из слов вида  $u_1 \dots u_r$ , где  $r \in \mathbb{N}$ ,  $u_1, \dots, u_r \in L$ .
- Например,  $(a|b)^* = \{\varepsilon, a, b, aa, ab, ba, bb, \dots\}$ .

# Регулярные выражения

Пусть зафиксирован конечный алфавит  $\Sigma$ .

- Базовые регулярные выражения: элементы алфавита,
- Также есть константы 0 (пустой язык) и 1 (язык, содержащий только пустое слово  $\varepsilon$ ).
- Бинарные операции:  $|$  (объединение) и  $\cdot$  (конкатенация или приписывание):  $u \cdot v = uv$ ,
- Унарная операция  $*$  (итерация, взять любое количество раз):  $L^*$  состоит из слов вида  $u_1 \dots u_r$ , где  $r \in \mathbb{N}$ ,  $u_1, \dots, u_r \in L$ .
- Например,  $(a|b)^* = \{\varepsilon, a, b, aa, ab, ba, bb, \dots\}$ .
- $(a|ab)^* = \{w \in \{a, b\}^+ | w = au, u \in \{a, b\}^+\}$

# Регулярные выражения

Пусть зафиксирован конечный алфавит  $\Sigma$ .

- Базовые регулярные выражения: элементы алфавита,
- Также есть константы 0 (пустой язык) и 1 (язык, содержащий только пустое слово  $\varepsilon$ ).
- Бинарные операции:  $|$  (объединение) и  $\cdot$  (конкатенация или приписывание):  $u \cdot v = uv$ ,
- Унарная операция  $*$  (итерация, взять любое количество раз):  $L^*$  состоит из слов вида  $u_1 \dots u_r$ , где  $r \in \mathbb{N}$ ,  $u_1, \dots, u_r \in L$ .
- Например,  $(a|b)^* = \{\varepsilon, a, b, aa, ab, ba, bb, \dots\}$ .
- $(a|ab)^* = \{w \in \{a, b\}^+ | w = au, u \in \{a, b\}^+\}$
- Приоритет операций: итерация, конкатенация, объединение.  
При этом значок конкатенации можно опускать.

# Регулярные выражения

Пусть зафиксирован конечный алфавит  $\Sigma$ .

- Базовые регулярные выражения: элементы алфавита,
- Также есть константы 0 (пустой язык) и 1 (язык, содержащий только пустое слово  $\varepsilon$ ).
- Бинарные операции:  $|$  (объединение) и  $\cdot$  (конкатенация или приписывание):  $u \cdot v = uv$ ,
- Унарная операция  $*$  (итерация, взять любое количество раз):  $L^*$  состоит из слов вида  $u_1 \dots u_r$ , где  $r \in \mathbb{N}$ ,  $u_1, \dots, u_r \in L$ .
- Например,  $(a|b)^* = \{\varepsilon, a, b, aa, ab, ba, bb, \dots\}$ .
- $(a|ab)^* = \{w \in \{a, b\}^+ | w = au, u \in \{a, b\}^+\}$
- Приоритет операций: итерация, конкатенация, объединение.  
При этом значок конкатенации можно опускать.

Язык регулярный, если он задаётся регулярным выражением.

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  
 $(a|b|c)^* b(a|b|c)$ .

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  
 $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  
 $(b|c)^* a(b|c)^* a(b|c)^*$ .

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  
 $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  
 $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие чётное число букв  $a$ :

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие чётное число букв  $a$ :  $((b|c)^* a(b|c)^* a(b|c)^*)^*$ .

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  
 $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  
 $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие чётное число букв  $a$ :  
 $((b|c)^* a(b|c)^* a)(b|c)^*$ .
- Слова в алфавите  $\{a, b, c\}$ , где перед  $a$  идёт только  $b$ :

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  
 $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  
 $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие чётное число букв  $a$ :  
 $((b|c)^* a(b|c)^* a)(b|c)^*$ .
- Слова в алфавите  $\{a, b, c\}$ , где перед  $a$  идёт только  $b$ :  
 $((b|c)^* ba)^* (b|c)^*$ .

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие чётное число букв  $a$ :  $((b|c)^* a(b|c)^* a)(b|c)^*$ .
- Слова в алфавите  $\{a, b, c\}$ , где перед  $a$  идёт только  $b$ :  $((b|c)^* ba)^* (b|c)^*$ .
- Непустые слова в алфавите  $\{a, b\}$ , в которых одинаковые буквы не идут подряд: .

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие чётное число букв  $a$ :  $((b|c)^* a(b|c)^* a)(b|c)^*$ .
- Слова в алфавите  $\{a, b, c\}$ , где перед  $a$  идёт только  $b$ :  $((b|c)^* ba)^* (b|c)^*$ .
- Непустые слова в алфавите  $\{a, b\}$ , в которых одинаковые буквы не идут подряд:  $(a(ba)^*(b|1))|(b(ab)^*(a|1))$ .

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие чётное число букв  $a$ :  $((b|c)^* a(b|c)^* a)(b|c)^*$ .
- Слова в алфавите  $\{a, b, c\}$ , где перед  $a$  идёт только  $b$ :  $((b|c)^* ba)^* (b|c)^*$ .
- Непустые слова в алфавите  $\{a, b\}$ , в которых одинаковые буквы не идут подряд:  $(a(ba)^*(b|1))|(b(ab)^*(a|1))$ .
- Непустые слова в алфавите  $\{a, b, c\}$ , в которых одинаковые буквы не идут подряд:

## Примеры регулярных языков

- Все слова в алфавите  $\{a, b\}$ :  $(a|b)^*$ ,
- Слова в алфавите  $\{a, b, c\}$ , где предпоследняя буква —  $b$ :  $(a|b|c)^* b(a|b|c)$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие ровно 2 буквы  $a$ :  $(b|c)^* a(b|c)^* a(b|c)^*$ .
- Слова нечётной длины в алфавите  $\{a, b\}$ :  $((a|b)(a|b))^*$ .
- Слова в алфавите  $\{a, b, c\}$ , содержащие чётное число букв  $a$ :  $((b|c)^* a(b|c)^* a)(b|c)^*$ .
- Слова в алфавите  $\{a, b, c\}$ , где перед  $a$  идёт только  $b$ :  $((b|c)^* ba)^* (b|c)^*$ .
- Непустые слова в алфавите  $\{a, b\}$ , в которых одинаковые буквы не идут подряд:  $(a(ba)^*(b|1))|(b(ab)^*(a|1))$ .
- Непустые слова в алфавите  $\{a, b, c\}$ , в которых одинаковые буквы не идут подряд:  $(H|1)(cH)^*(1|c|cH)$ , где  $H$  — ответ на предыдущий пункт.

## Примеры регулярных выражений

Пусть  $\Sigma = \{C, V, \overline{V}, -\}$  (согласный, безударный гласный, ударный гласный, слогораздел).

- Корректное разбиение на слоги:
  - В каждом слоге ровно одна гласная:  $C^*(V|\overline{V})C^*$ .
  - Ровно один слог ударный.

## Примеры регулярных выражений

Пусть  $\Sigma = \{C, V, \overline{V}, -\}$  (согласный, безударный гласный, ударный гласный, слогораздел).

- Корректное разбиение на слоги:
  - В каждом слоге ровно одна гласная:  $C^*(V|\overline{V})C^*$ .
  - Ровно один слог ударный.
  - Пусть  $X$  — ударный слог,  $Y$  — безударный, тогда искомое выражение  $(Y-)^*[(X - (Y-)^*Y)|X]$ .

## Примеры регулярных выражений

Пусть  $\Sigma = \{C, V, \bar{V}, -\}$  (согласный, безударный гласный, ударный гласный, слогораздел).

- Корректное разбиение на слоги:
  - В каждом слоге ровно одна гласная:  $C^*(V|\bar{V})C^*$ .
  - Ровно один слог ударный.
  - Пусть  $X$  — ударный слог,  $Y$  — безударный, тогда искомое выражение  $(Y-)^*[(X-(Y-)^*Y)|X]$ .
  - Эквивалентно  $(Y-)^*X(Y-)^* = (C^*VC^*-)^*C^*\bar{V}C^*(-C^*VC^*)^*$ .

## Примеры регулярных выражений

Пусть  $\Sigma = \{C, V, \bar{V}, -\}$  (согласный, безударный гласный, ударный гласный, слогораздел).

- Корректное разбиение на слоги:
  - В каждом слоге ровно одна гласная:  $C^*(V|\bar{V})C^*$ .
  - Ровно один слог ударный.
  - Пусть  $X$  — ударный слог,  $Y$  — безударный, тогда искомое выражение  $(Y-)^*[(X-(Y-)^*Y)|X]$ .
  - Эквивалентно  $(Y-)^*X(Y-)^* = (C^*VC^*-)^*C^*\bar{V}C^*(-C^*VC^*)^*$ .
- Разбиение слова на слоги, содержащее ровно 1 открытый слог (ударность не учитывается).

## Примеры регулярных выражений

Пусть  $\Sigma = \{C, V, \bar{V}, -\}$  (согласный, безударный гласный, ударный гласный, слогораздел).

- Корректное разбиение на слоги:
  - В каждом слоге ровно одна гласная:  $C^*(V|\bar{V})C^*$ .
  - Ровно один слог ударный.
  - Пусть  $X$  — ударный слог,  $Y$  — безударный, тогда искомое выражение  $(Y-)^*[(X - (Y-)^*Y)|X]$ .
  - Эквивалентно  $(Y-)^*X(Y-)^* = (C^*VC^*-)^*C^*\bar{V}C^*(-C^*VC^*)^*$ .
- Разбиение слова на слоги, содержащее ровно 1 открытый слог (ударность не учитывается).  $(C^*VC^+-)^*(C^*V)(-C^*VC^+)^*$

## Примеры регулярных выражений

Пусть  $\Sigma = \{C, V, \bar{V}, -\}$  (согласный, безударный гласный, ударный гласный, слогораздел).

- Корректное разбиение на слоги:
  - В каждом слоге ровно одна гласная:  $C^*(V|\bar{V})C^*$ .
  - Ровно один слог ударный.
  - Пусть  $X$  — ударный слог,  $Y$  — безударный, тогда искомое выражение  $(Y-)^*[(X - (Y-)^*Y)|X]$ .
  - Эквивалентно  $(Y-)^*X(Y-)^* = (C^*VC^*-)^*C^*\bar{V}C^*(-C^*VC^*)^*$ .
- Разбиение слова на слоги, содержащее ровно 1 открытый слог (ударность не учитывается).  $(C^*VC^+-)^*(C^*V)(-C^*VC^+)^*$
- “Гармония гласных” (гласные типа  $V_1$  и  $V_2$  не встречаются вместе):  $(C|V)^*(V_1(C|V_1|V)^*|V_2(C|V_2|V)^*)$

## Примеры регулярных выражений

- Множественное число существительного в английском:
  - *-es* после шипящих (*s*, *x*, *z*, *ch*, *sh*, *zh*).
  - *-y* после согласных перед *-s* переходит в *-ie*.

## Примеры регулярных выражений

- Множественное число существительного в английском:
  - -es после шипящих (*s*, *x*, *z*, *ch*, *sh*, *zh*).
  - -у после согласных перед -s переходит в -ie.
- Удобней разбирать *witches* = *witche*+*s*, *enemies* = *enemie*+*s*.

## Примеры регулярных выражений

- Множественное число существительного в английском:
  - *-es* после шипящих (*s*, *x*, *z*, *ch*, *sh*, *zh*).
  - *-u* после согласных перед *-s* переходит в *-ie*.
- Удобней разбирать *witches = witche+s*, *enemies = enemy+s*.
- Искомое выражение  $X = Ys$ , где  $Y$  — выражение для основы.

## Примеры регулярных выражений

- Множественное число существительного в английском:
  - -es после шипящих (*s*, *x*, *z*, *ch*, *sh*, *zh*).
  - -у после согласных перед -s переходит в -ie.
- Удобней разбирать *witches* = *witch*+*s*, *enemies* = *enemie*+*s*.
- Искомое выражение  $X = Ys$ , где  $Y$  — выражение для основы.
- Основа — всё, что не кончается на *s*, *x*, *z*, *ch*, *sh*, *zh*, **Cy**.

## Примеры регулярных выражений

- Множественное число существительного в английском:
  - -es после шипящих (*s*, *x*, *z*, *ch*, *sh*, *zh*).
  - -у после согласных перед -s переходит в -ie.
- Удобней разбирать *witches* = *witche*+*s*, *enemies* = *enemie*+*s*.
- Искомое выражение  $X = Ys$ , где  $Y$  — выражение для основы.
- Основа — всё, что не кончается на *s*, *x*, *z*, *ch*, *sh*, *zh*, **Cy**.
- Хочется задать отрицание условия...

## Примеры регулярных выражений

- Множественное число существительного в английском:
  - -es после шипящих (*s*, *x*, *z*, *ch*, *sh*, *zh*).
  - -у после согласных перед -s переходит в -ie.
- Удобней разбирать *witches* = *witche*+*s*, *enemies* = *enemie*+*s*.
- Искомое выражение  $X = Ys$ , где  $Y$  — выражение для основы.
- Основа — всё, что не кончается на *s*, *x*, *z*, *ch*, *sh*, *zh*, **Cy**.
- Хочется задать отрицание условия...
- Симулируется через перечисление.

## Примеры регулярных выражений

- Корректная основа:

## Примеры регулярных выражений

- Корректная основа:
  - Заканчивается на гласный, не равный у:  $(C|V)^*(a|e|i|o|u)$ .

## Примеры регулярных выражений

- Корректная основа:
  - Заканчивается на гласный, не равный у:  $(C|V)^*(a|e|i|o|u)$ .
  - Заканчивается на гласный+у:  $(C|V)^*Vu$ .

## Примеры регулярных выражений

- Корректная основа:
  - Заканчивается на гласный, не равный  $y$ :  $(C|V)^*(a|e|i|o|u)$ .
  - Заканчивается на гласный+ $y$ :  $(C|V)^*Vy$ .
  - Содержит гласный и заканчивается на согласный, но не на  $s, x, z, h$  ( $C'$  — полный список таких согласных):  
 $(C|V)^*V(C|V)^*C'$

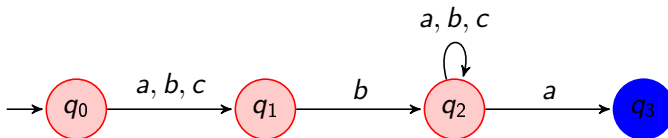
# Примеры регулярных выражений

- Корректная основа:
  - Заканчивается на гласный, не равный  $y$ :  $(C|V)^*(a|e|i|o|u)$ .
  - Заканчивается на гласный  $+y$ :  $(C|V)^*Vy$ .
  - Содержит гласный и заканчивается на согласный, но не на  $s, x, z, h$  ( $C'$  — полный список таких согласных):  
 $(C|V)^*V(C|V)^*C'$
  - Содержит гласный и заканчивается на  $h$  или  $C''h$ , где  $C''$  обозначает любой согласный, кроме  $s, c, h$ :  
 $(C|V)^*V((C|V)^*C'')?h$

## Примеры регулярных выражений

- Корректная основа:
  - Заканчивается на гласный, не равный  $y$ :  $(C|V)^*(a|e|i|o|u)$ .
  - Заканчивается на гласный  $+y$ :  $(C|V)^*Vy$ .
  - Содержит гласный и заканчивается на согласный, но не на  $s, x, z, h$  ( $C'$  — полный список таких согласных):  
 $(C|V)^*V(C|V)^*C'$
  - Содержит гласный и заканчивается на  $h$  или  $C''h$ , где  $C''$  обозначает любой согласный, кроме  $s, c, h$ :  
 $(C|V)^*V((C|V)^*C'')?h$
- Всё вместе:  $(C|V)^*((a|e|i|o|u|Vy) | V(h|(C|V)^*(C'|C''h))s$ .

## Определение конечного автомата



- Данная картинка описывает алгоритм, принимающий все слова  $w$ , где  $w[1] = b$  и  $w[-1] = a$ .
- Принимаемые слова — метки путей из стартового состояния в завершающее.
- Как это задать формально?

# Определение конечного автомата

Пусть  $\Sigma$  — конечный алфавит.

## Определение конечного автомата

Конечный автомат: помеченный граф  $M = \langle Q, \Sigma, \Delta, q_0, F \rangle$ , где

# Определение конечного автомата

Пусть  $\Sigma$  — конечный алфавит.

## Определение конечного автомата

Конечный автомат: помеченный граф  $M = \langle Q, \Sigma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний (вершин графа),

# Определение конечного автомата

Пусть  $\Sigma$  — конечный алфавит.

## Определение конечного автомата

Конечный автомат: помеченный граф  $M = \langle Q, \Sigma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний (вершин графа),
- $\Delta$  — конечное множество переходов (рёбер с метками) вида  $\langle q_1, w \rangle \rightarrow q_2, q_1, q_2 \in Q, w \in \Sigma^*$ .

# Определение конечного автомата

Пусть  $\Sigma$  — конечный алфавит.

## Определение конечного автомата

Конечный автомат: помеченный граф  $M = \langle Q, \Sigma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний (вершин графа),
- $\Delta$  — конечное множество переходов (рёбра с метками) вида  $\langle q_1, w \rangle \rightarrow q_2, q_1, q_2 \in Q, w \in \Sigma^*$ .
- $q_0 \in Q$  — стартовое состояние
- $F \subseteq Q$  — завершающие состояния.

# Определение конечного автомата

Пусть  $\Sigma$  — конечный алфавит.

## Определение конечного автомата

Конечный автомат: помеченный граф  $M = \langle Q, \Sigma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний (вершин графа),
- $\Delta$  — конечное множество переходов (рёбер с метками) вида  $\langle q_1, w \rangle \rightarrow q_2, q_1, q_2 \in Q, w \in \Sigma^*$ .
- $q_0 \in Q$  — стартовое состояние
- $F \subseteq Q$  — завершающие состояния.

Метка пути — конкатенация всех меток на входящих в него рёбрах.

# Определение конечного автомата

Пусть  $\Sigma$  — конечный алфавит.

## Определение конечного автомата

Конечный автомат: помеченный граф  $M = \langle Q, \Sigma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний (вершин графа),
- $\Delta$  — конечное множество переходов (рёбер с метками) вида  $\langle q_1, w \rangle \rightarrow q_2, q_1, q_2 \in Q, w \in \Sigma^*$ .
- $q_0 \in Q$  — стартовое состояние
- $F \subseteq Q$  — завершающие состояния.

Метка пути — конкатенация всех меток на входящих в него рёбрах.

Язык  $L(M)$  — метки путей из начального состояния в завершающие.

# Определение конечного автомата

Пусть  $\Sigma$  — конечный алфавит.

## Определение конечного автомата

Конечный автомат: помеченный граф  $M = \langle Q, \Sigma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний (вершин графа),
- $\Delta$  — конечное множество переходов (рёбра с метками) вида  $\langle q_1, w \rangle \rightarrow q_2, q_1, q_2 \in Q, w \in \Sigma^*$ .
- $q_0 \in Q$  — стартовое состояние
- $F \subseteq Q$  — завершающие состояния.

Метка пути — конкатенация всех меток на входящих в него рёбрах.

Язык  $L(M)$  — метки путей из начального состояния в завершающие.

Язык — автоматный, если задаётся некоторым конечным автоматом.

# Примеры конечных автоматов

- Закрытый слог

# Примеры конечных автоматов

- Закрытый слог



## Примеры конечных автоматов

- Закрытый слог



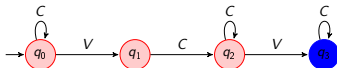
- Слово с 2 гласными, разделёнными хотя бы одним согласным:

## Примеры конечных автоматов

- Закрытый слог



- Слово с 2 гласными, разделёнными хотя бы одним согласным:

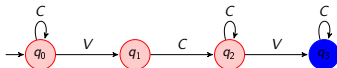


## Примеры конечных автоматов

- Закрытый слог



- Слово с 2 гласными, разделёнными хотя бы одним согласным:



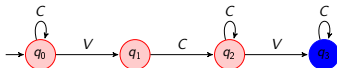
- Слогоделение ровно с одним открытым слогом:

## Примеры конечных автоматов

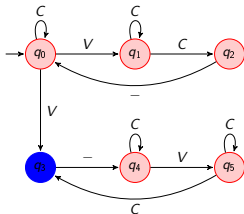
- Закрытый слог



- Слово с 2 гласными, разделёнными хотя бы одним согласным:



- Слогоделение ровно с одним открытым слогом:

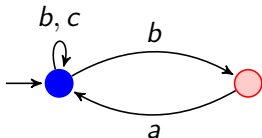


## Конечные автоматы: примеры

- Каждой  $a$  непосредственно предшествует  $b$ , алфавит  $a, b, c$ .

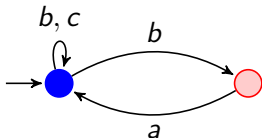
## Конечные автоматы: примеры

- Каждой  $a$  непосредственно предшествует  $b$ , алфавит  $a, b, c$ .



## Конечные автоматы: примеры

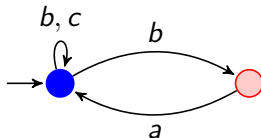
- Каждой  $a$  непосредственно предшествует  $b$ , алфавит  $a, b, c$ .



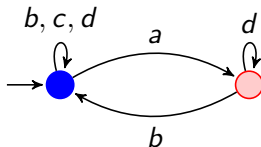
- Справа от каждой  $a$  есть парная ей  $b$ , между парными буквами нет  $a, c$ , алфавит  $a, b, c, d$ .

## Конечные автоматы: примеры

- Каждой  $a$  непосредственно предшествует  $b$ , алфавит  $a, b, c$ .

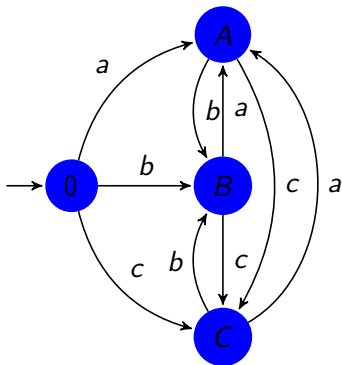


- Справа от каждой  $a$  есть парная ей  $b$ , между парными буквами нет  $a, c$ , алфавит  $a, b, c, d$ .



## Конечные автоматы: примеры

Нет повторяющихся букв, алфавит  $a, b, c$ . Состояния соответствуют буквам:



## Конечные автоматы: примеры

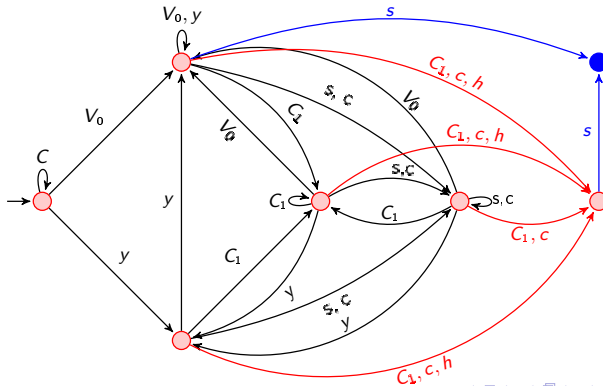
- Формы множественного числа представимы в виде  $stem + s$ , где

## Конечные автоматы: примеры

- Формы множественного числа представимы в виде  $\text{stem} + s$ , где
- $\text{stem}$  обязательно содержит гласную и не кончается на:
  - $-s, -x, -z, -sh, -ch, -zh$  (шипящие).
  - $Cy$ .

## Конечные автоматы: примеры

- Формы множественного числа представимы в виде  $\text{stem} + s$ , где
- $\text{stem}$  обязательно содержит гласную и не кончается на:
  - $-s, -x, -z, -sh, -ch, -zh$  (шипящие).
  - $Cy$ .
- Автомат для основ  
 $(C_0 = C - \{s, x, z, c, h\}, C_1 = C_0 \cup \{s, x, z\})$ :



## Свойства автоматных языков

### Теорема

*Любой автоматный язык распознаётся автоматом с однобуквенными переходами.*

# Свойства автоматных языков

## Теорема

*Любой автоматный язык распознаётся автоматом с однобуквенными переходами.*

## Определение

*Автомат с однобуквенными переходами — детерминированный, если ни из какого состояния нет двух переходов по одной и той же букве.*

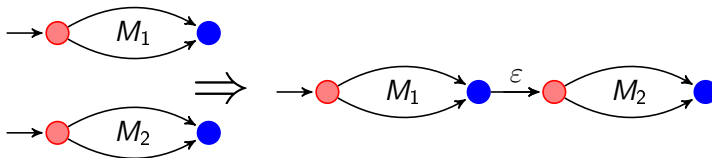
## Теорема

*Любой автоматный язык распознаётся детерминированным конечным автоматом.*

# Свойства автоматных языков

Автоматные языки замкнуты относительно:

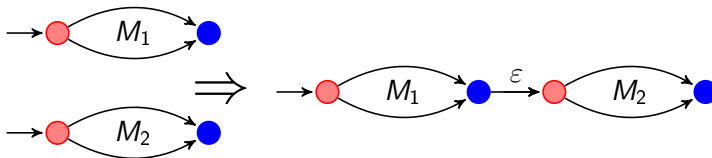
- Конкатенации.



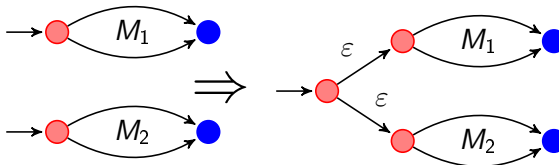
# Свойства автоматных языков

Автоматные языки замкнуты относительно:

- Конкатенации.



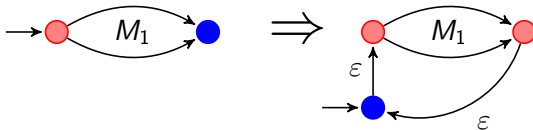
- Объединения:



# Свойства автоматных языков

Автоматные языки замкнуты относительно:

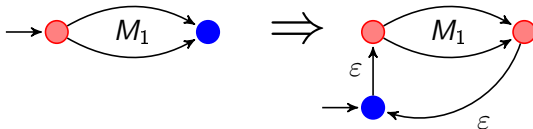
- Итерации:



## Свойства автоматных языков

Автоматные языки замкнуты относительно:

- Итерации:

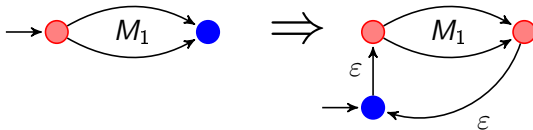


- Следствие: всякий регулярный язык является автоматным.

## Свойства автоматных языков

Автоматные языки замкнуты относительно:

- Итерации:



- Следствие: всякий регулярный язык является автоматным.
- На самом деле верно и обратное утверждение (теорема Клини): всякий автоматный язык регулярен.

# Рекурсивное построение автоматов

- Автоматные языки замкнуты относительно:
  - Конкатенации.
  - Объединения.
  - Итерации.

# Рекурсивное построение автоматов

- Автоматные языки замкнуты относительно:
  - Конкатенации.
  - Объединения.
  - Итерации.
  - Дополнения.
  - Пересечения.

# Рекурсивное построение автоматов

- Автоматные языки замкнуты относительно:
  - Конкатенации.
  - Объединения.
  - Итерации.
  - Дополнения.
  - Пересечения.
- При этом соответствующие автоматы строятся алгоритмически.

# Рекурсивное построение автоматов

- Автоматные языки замкнуты относительно:
  - Конкатенации.
  - Объединения.
  - Итерации.
  - Дополнения.
  - Пересечения.
- При этом соответствующие автоматы строятся алгоритмически.
- Вывод: автоматы можно строить поэтапно, “собирая” из простых сложные с помощью различных операций.

# Рекурсивное построение автоматов

- Множественное число в английском:

$$(L_{sib} \cdot es) \cup (((\overline{L_{sib}} \cap L_C) \cup L_{Vy} \cup L_V) \cdot s),$$

где

- $L_{sib}$  — слова, кончающиеся на шипящий.
- $L_C$  — слова, кончающиеся на согласный.
- $L_{Vy}$  — слова кончающиеся на гласный + у.
- $L_V$  — слова, кончающиеся на гласный (не у).

# Рекурсивное построение автоматов

- Множественное число в английском:

$$(L_{sib} \cdot es) \cup (((\overline{L_{sib}} \cap L_C) \cup L_{Vy} \cup L_V) \cdot s),$$

где

- $L_{sib}$  — слова, кончающиеся на шипящий.
  - $L_C$  — слова, кончающиеся на согласный.
  - $L_{Vy}$  — слова кончающиеся на гласный + у.
  - $L_V$  — слова, кончающиеся на гласный (не у).
- $L_{sib}, L_C, L_{Vy}, L_V$  — автоматные, финальный автомат строится рекурсивно.

# Рекурсивное построение автомата

## Инфинитив глагола в турецком

Построить конечный автомат для инфинитива в турецком

- Инфинитив имеет форму основа+ $mEk$ .
- $E$  равно  $e$ , если последний гласный основы —  $e, i, ö, ü$ , и  $a$  — если  $a, ı, o, u$ .

# Рекурсивное построение автомата

## Инфинитив глагола в турецком

Построить конечный автомат для инфинитива в турецком

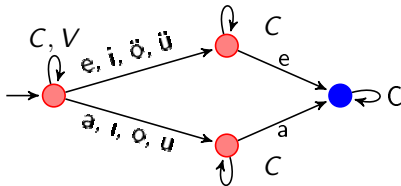
- Инфинитив имеет форму основа+ $mEk$ .
- $E$  равно  $e$ , если последний гласный основы —  $e, i, ö, ü$ , и  $a$  — если  $a, ı, o, u$ .
- $M_1$  — автомат для выражения  $C^*V(C|V)^*m(a|e)k$ .

# Рекурсивное построение автомата

## Инфинитив глагола в турецком

Построить конечный автомат для инфинитива в турецком

- Инфинитив имеет форму основа+ $mEk$ .
- $E$  равно  $e$ , если последний гласный основы —  $e, i, ö, ü$ , и  $a$  — если  $a, ı, o, u$ .
- $M_1$  — автомат для выражения  $C^*V(C|V)^*m(a|e)k$ .
- $M_2$  проверяет гармонию гласных:



- $M_1 \cap M_2$  — требуемый автомат.

# Конечные преобразователи

Неформально, конечный преобразователь — это автомат с добавленными на рёбра выходными символами.

## Конечные преобразователи

Неформально, конечный преобразователь — это автомат с добавленными на рёбра выходными символами.

Пусть  $\Sigma$ ,  $\Gamma$  — конечные алфавиты.

### Определение конечного преобразователя

Конечный преобразователь: кортеж  $M = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний

## Конечные преобразователи

Неформально, конечный преобразователь — это автомат с добавленными на рёбра выходными символами.

Пусть  $\Sigma$ ,  $\Gamma$  — конечные алфавиты.

### Определение конечного преобразователя

Конечный преобразователь: кортеж  $M = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний
- $\Delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$  — конечное множество переходов

## Конечные преобразователи

Неформально, конечный преобразователь — это автомат с добавленными на рёбра выходными символами.

Пусть  $\Sigma$ ,  $\Gamma$  — конечные алфавиты.

### Определение конечного преобразователя

Конечный преобразователь: кортеж  $M = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний
- $\Delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$  — конечное множество переходов
- $q_0 \in Q$  — стартовое состояние
- $F \subseteq Q$  — завершающие состояния.

## Конечные преобразователи

Неформально, конечный преобразователь — это автомат с добавленными на рёбра выходными символами.

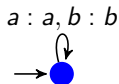
Пусть  $\Sigma$ ,  $\Gamma$  — конечные алфавиты.

### Определение конечного преобразователя

Конечный преобразователь: кортеж  $M = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний
- $\Delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$  — конечное множество переходов
- $q_0 \in Q$  — стартовое состояние
- $F \subseteq Q$  — завершающие состояния.

Простейший преобразователь — тождественный (алфавит  $a, b$ ):



## Конечные преобразователи

Неформально, конечный преобразователь — это автомат с добавленными на рёбра выходными символами.

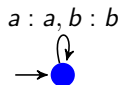
Пусть  $\Sigma$ ,  $\Gamma$  — конечные алфавиты.

### Определение конечного преобразователя

Конечный преобразователь: кортеж  $M = \langle Q, \Sigma, \Gamma, \Delta, q_0, F \rangle$ , где

- $Q$  — конечное множество состояний
- $\Delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \times Q$  — конечное множество переходов
- $q_0 \in Q$  — стартовое состояние
- $F \subseteq Q$  — завершающие состояния.

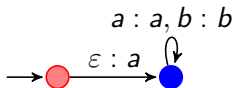
Простейший преобразователь — тождественный (алфавит  $a, b$ ):



Конечные автоматы можно понимать как преобразователи, возвращающие свой вход (и проверяющие, что вход принадлежит нужному множеству).

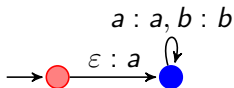
# Конечные преобразователи: примеры

- Добавляет  $a$  в начале слова:

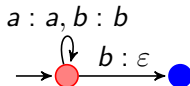


## Конечные преобразователи: примеры

- Добавляет  $a$  в начале слова:

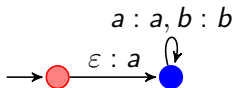


- Удаляет конечную  $b$  в тех словах, где она есть, и отвергает остальные:

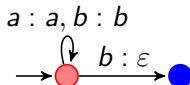


## Конечные преобразователи: примеры

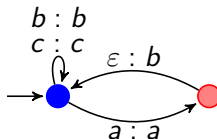
- Добавляет  $a$  в начале слова:



- Удаляет конечную  $b$  в тех словах, где она есть, и отвергает остальные:

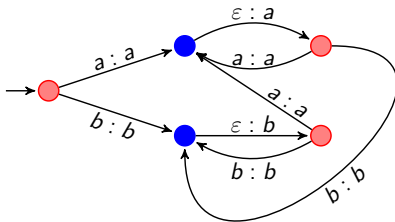


- После каждой  $a$  добавляет  $b$ :



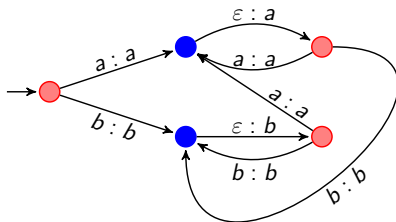
## Конечные преобразователи: примеры

- Удваивает все буквы, кроме последней:

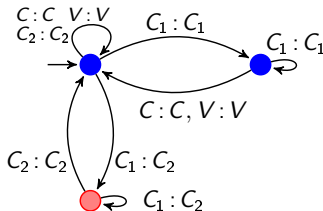


## Конечные преобразователи: примеры

- Удваивает все буквы, кроме последней:



- Ретро-ассимилирует  $C_1$  в  $C_2$  (в последовательности  $C_1$ , идущей перед  $C_2$ , все буквы заменяются на  $C_2$ )



## Свойства конечных преобразователей

- Замкнутость относительно операций:

| Операция     | Автоматы | Преобразователи |
|--------------|----------|-----------------|
| Конкатенация | Да       | Да              |
| Итерация     | Да       | Да              |
| Объединение  | Да       | Да              |
| Пересечение  | Да       | Нет             |
| Дополнение   | Да       | Нет             |

# Свойства конечных преобразователей

- Конечные преобразования также замкнуты относительно
  - Композиции ( $\circ$ ).
  - Приоритетного объединения ( $\cup_p$ ).

## Свойства конечных преобразователей

- Конечные преобразования также замкнуты относительно
  - Композиции ( $\circ$ ).
  - Приоритетного объединения ( $\cup_p$ ).

$$(T_1 \cup_p T_2)(u) = \begin{cases} T_1(u), & T_1(u) \neq \emptyset, \\ T_2(u), & T_1(u) = \emptyset. \end{cases}$$

## Свойства конечных преобразователей

- Конечные преобразования также замкнуты относительно
  - Композиции ( $\circ$ ).
  - Приоритетного объединения ( $\cup_p$ ).

$$(T_1 \cup_p T_2)(u) = \begin{cases} T_1(u), & T_1(u) \neq \emptyset, \\ T_2(u), & T_1(u) = \emptyset. \end{cases}$$

- Обращения  $((\cdot)^{-1})$ :  $\phi^{-1} = \{\langle y, x \rangle \mid \langle x, y \rangle \in \phi\}$ .

## Свойства конечных преобразователей

- Конечные преобразования также замкнуты относительно
  - Композиции ( $\circ$ ).
  - Приоритетного объединения ( $\cup_p$ ).

$$(T_1 \cup_p T_2)(u) = \begin{cases} T_1(u), & T_1(u) \neq \emptyset, \\ T_2(u), & T_1(u) = \emptyset. \end{cases}$$

- Обращения  $((\cdot)^{-1})$ :  $\phi^{-1} = \{\langle y, x \rangle \mid \langle x, y \rangle \in \phi\}$ .
- Применения операций:
  - Композиция: последовательное применение преобразований,

## Свойства конечных преобразователей

- Конечные преобразования также замкнуты относительно
  - Композиции ( $\circ$ ).
  - Приоритетного объединения ( $\cup_p$ ).

$$(T_1 \cup_p T_2)(u) = \begin{cases} T_1(u), & T_1(u) \neq \emptyset, \\ T_2(u), & T_1(u) = \emptyset. \end{cases}$$

- Обращения  $((\cdot)^{-1})$ :  $\phi^{-1} = \{\langle y, x \rangle \mid \langle x, y \rangle \in \phi\}$ .
- Применения операций:
  - Композиция: последовательное применение преобразований,
  - Обращение: переход от синтеза к анализу и наоборот,

## Свойства конечных преобразователей

- Конечные преобразования также замкнуты относительно
  - Композиции ( $\circ$ ).
  - Приоритетного объединения ( $\cup_p$ ).

$$(T_1 \cup_p T_2)(u) = \begin{cases} T_1(u), & T_1(u) \neq \emptyset, \\ T_2(u), & T_1(u) = \emptyset. \end{cases}$$

- Обращения  $((\cdot)^{-1})$ :  $\phi^{-1} = \{\langle y, x \rangle \mid \langle x, y \rangle \in \phi\}$ .
- Применения операций:
  - Композиция: последовательное применение преобразований,
  - Обращение: переход от синтеза к анализу и наоборот,
  - Приоритетное объединение: отдельная обработка нерегулярных форм.

# Множественное число существительного в английском

## Пример.

Опишите преобразователь, преобразующий форму единственного числа существительного в форму множественного для английского языка.

# Множественное число существительного в английском

## Пример.

Опишите преобразователь, преобразующий форму единственного числа существительного в форму множественного для английского языка.

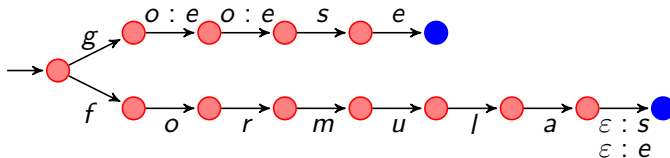
- $torch \leftrightarrow torches$
- $monarch \leftrightarrow monarchs$
- $ally \leftrightarrow allies$
- $play \leftrightarrow plays$
- $goose \leftrightarrow geese$
- $formula \leftrightarrow formulas/formulae$

# Пример: множественное число в английском

## Пример.

Описать преобразователь, строящий форму множественного числа существительного в английском фзыке.

- Отдельный преобразователь  $T_{exc}$  для исключений:

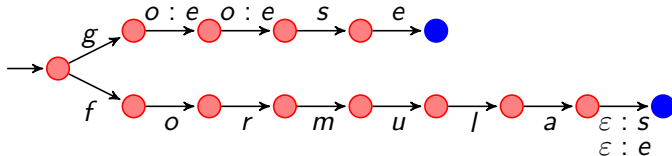


## Пример: множественное число в английском

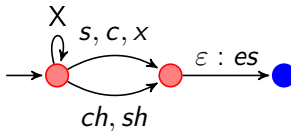
### Пример.

Описать преобразователь, строящий форму множественного числа существительного в английском фзыке.

- Отдельный преобразователь  $T_{exc}$  для исключений:

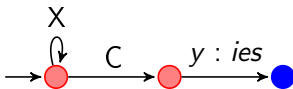


- $T_{sib}$ : добавляет -es после финальной шипящей ( $X$  — произвольный символ):



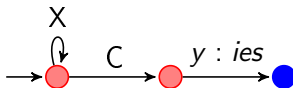
## Пример: множественное число в английском

- Преобразователь  $T_{exc}$  для исключений.
- Преобразователь  $T_{sib}$  для добавления *es*.
- Преобразователь  $T_{Cy}$  для замены *y* на *-ies* после согласной.



## Пример: множественное число в английском

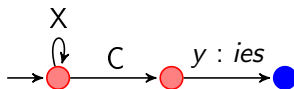
- Преобразователь  $T_{exc}$  для исключений.
- Преобразователь  $T_{sib}$  для добавления *es*.
- Преобразователь  $T_{Cy}$  для замены *y* на *-ies* после согласной.



- $T_s$  — добавляет *s* в конце.
- $T_{exc,sib}$  — добавляет *s* к исключениям на *-arch* и отвергает остальные слова (для *monarchs*, *tetrarchs*, ...).

## Пример: множественное число в английском

- Преобразователь  $T_{exc}$  для исключений.
- Преобразователь  $T_{sib}$  для добавления *es*.
- Преобразователь  $T_{Cy}$  для замены *y* на *-ies* после согласной.



- $T_s$  — добавляет *s* в конце.
- $T_{exc,sib}$  — добавляет *s* к исключениям на *-arch* и отвергает остальные слова (для *monarchs*, *tetrarchs*, ...).
- Общее решение:

$$T_{exc} \cup_p T_{exc,sib} \cup_p T_{sib} \cup_p T_{Cy} \cup_p T_s$$