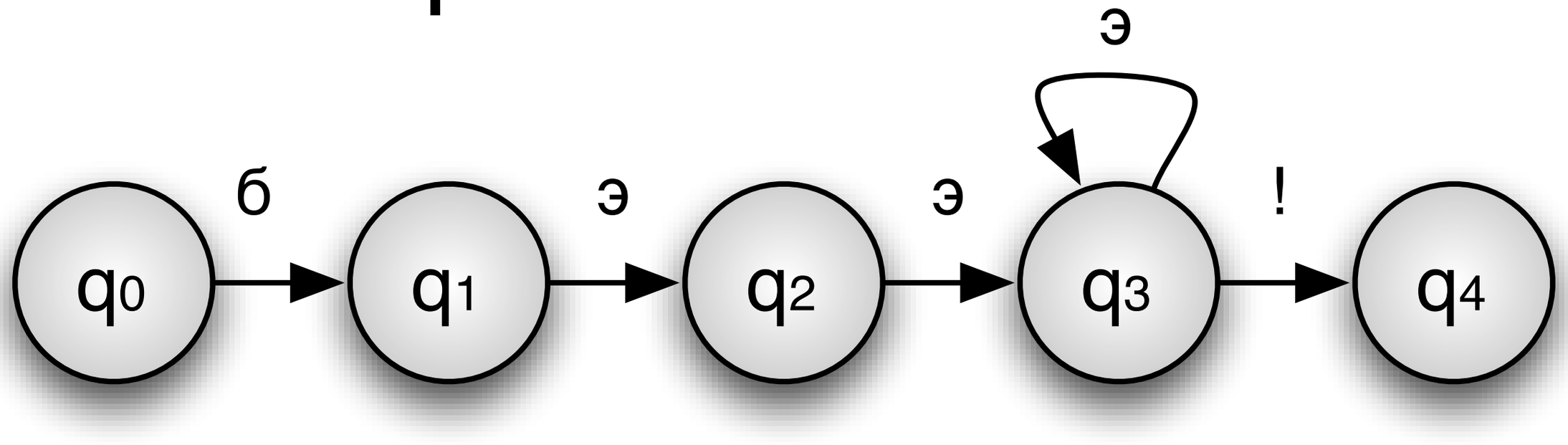
Введение в обработку текстов

Лекция 2

Регулярные выражения и конечные автоматы

Формальные языки

Формальные языки



- **формальный язык** — это множество конечных слов (строк, цепочек) над конечным алфавитом

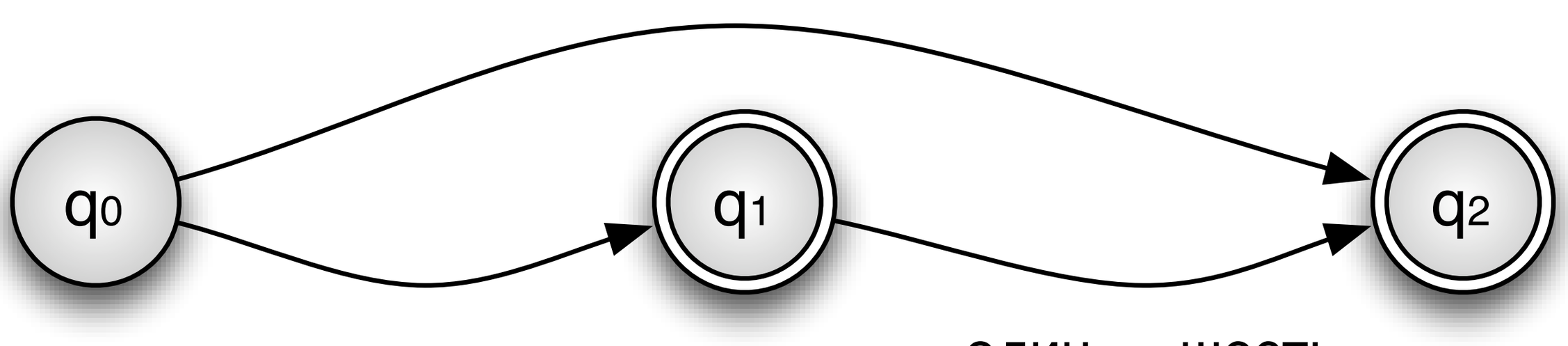
$$\Sigma = \{a, b, !\}$$
$$L(m) = \{baa!, baaa!, baaaa!, \ldots\}$$

Пример формального языка

один	шесть
два	семь
три	восемь
четыре	девять
пять	десять

одиннадцать
двенадцать
тринадцать
четырнадцать

пятнадцать
шестнадцать
семнадцать
восемнадцать
девятнадцать



двадцать	шестьдесят
тридцать	семьдесят
сорок	восемьдесят
пятьдесят	девяносто

один	шесть
два	семь
три	восемь
четыре	девять
пять	

Формальные языки

- Иерархия Хомского
  - Тип 0 – неограниченные
  - Тип 1 – контекстно-зависимые
  - Тип 2 – контекстно-свободные
  - Тип 3 – **регулярные**

Регулярные выражения

Мотивация

- Обновить цену товара в прайс-листе:

  для товаров за 1000р. сделать 999.99р.

- Заменить все вхождения одного слова в тексте на другое
  - для части слова (Википедия -> Энциклопедия)
  - с учетом контекста
- Найти сообщения о терроризме
- Фильтрация нецензурных высказываний на форумах

Регулярные выражения

- Regular Expressions (RegExp)
- Языки программирования (Python, Perl, Ruby, Java, .Net)
- Текстовые редакторы (Vim, EmEdit)
- Утилиты (grep, sed)

Регулярные выражения

- Регулярные выражения - алгебраическая нотация для записи множества строк
- Функции Python

```
#encoding=CP1251
import re
re.search("в","пиво").group(0) # в
re.sub("о","ко","пиво") # пивко
re.findall("cd","abcdcde") # [“cd”,”cd”]
```

Регулярные выражения

- Последовательность букв: *abcd*
- Чувствительны к регистру: *“Пиво”≠ “пиво”*
- Дизъюнкция: *[П|п]иво,[abc],[1234567890]*
- Интервал: *[A-Z], [0-9], [A-Za-z]*

```
for letter in re.findall("[а-о]","пиво"):
    print letter,
> и в о
```

- Знак ^: *[^a]* - все кроме *“a”*
- “.” - любой символ, кроме \n

Регулярные выражения

- ? - условие для 0 или 1 вхождения символа

```
re.findall("пивк?о", "пиво или пивко")
> [“пиво”,”пивко”]
```

- Как найти “Goooooogle”?
- Счетчики
  - Gooo*gle
  - Goo+gle

```
print re.sub("Goo+gle","Google","Goooooogle")
> Google
```

Регулярные выражения

- Дизъюнкция

```
re.sub("cat|dog","chicken","cat dog")
> chicken chicken
re.search("пив(о|ко)","пивко") # Match
```

- Якоря

^ - начало строки

$ - конец строки

```
re.search("^cat(1|2)","cat1 cat2").group(0)
> cat1
re.search("cat(1|2)$","cat1 cat2").group(0)
> cat 2
```

Регулярные выражения

- Память (Memory)

```
text="A conditional random field (CRF) is a ..."
print re.search("\([^\)]+\)",text).group(0)
> (CRF)
print re.search("\(([^\)]+)\)",text).group(1)
> CRF
```

```
text1 = "the faster they ran, the faster we ran"
text2 = "the faster they ran, the faster we ate"
re.search("the (.*)er they (.*), the \\1er we \\2", text1) # Match
re.search("the (.*)er they (.*), the \\1er we \\2", text2) # Not match
```

- Приоритет операций

Круглые скобки	()
Счетчики	* + ? {}
Последовательности и якоря	the ^my end$
Дизъюнкция	|

Практические примеры

- Найти все вхождения слова в тексте
  - слово “the” в тексте 7 раз

Практические примеры

- ELIZA

User: Men are all alike.
ELIZA: IN WHAT WAY?
User: They’re always bugging us about something other
ELIZA: CAN YOU THINK OF A SPECIFIC EXAMPLE?
User: My boyfriend made me come here
ELIZA: YOUR BOYFRIEND MADE YOU COME HERE
User: He says I’m depressed much of the time
ELIZA: I AM SORRY TO HEAR YOU ARE DEPRESSED

```
sentence=re.sub("I’m","YOU ARE", sentence)
...
sentence=re.sub(".* YOU ARE (depressed|sad) .*","I'M SORRY TO HEAR YOU ARE \\1",sentence)
sentence=re.sub(".* all .*","IN WHAT WAY?",sentence)
sentence=re.sub(".* always .*","CAN YOU THINK OF A SPECIFIC EXAMPLE?",sentence)
```

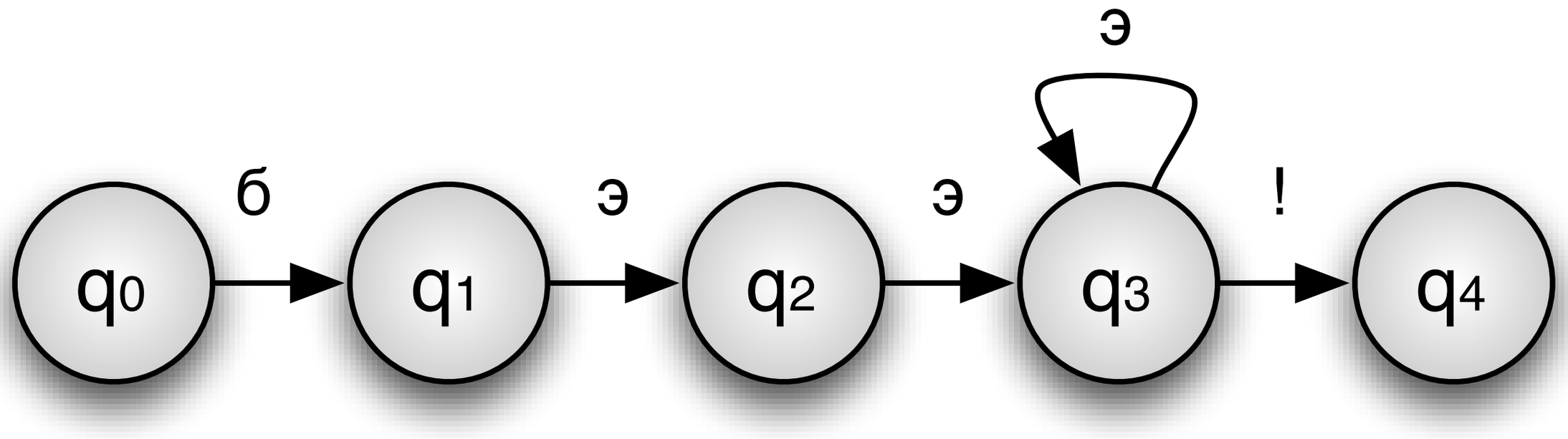
Конечные автоматы

Конечные автоматы

- Finite-state automation (FSA)
- Один из важнейших инструментов для обработки текстов
- Могут быть использованы для реализации регулярных выражений

Использование КА для распознавания языка

- Научимся говорить с овцами
  - –бээ!
  - –бээээ!
  - –бээээээ!
  - –бээээээээ!
  - –…
- RE: “бээ+!”



Представление автоматов

- Текст: лента с ячейками

- Таблица переходов между состояниями

Состояние	Вход		
	б	э	!
0	1	∅	∅
1	∅	2	∅
2	∅	3	∅
3	∅	3	4
4	∅	∅	∅

Формальное определение

$Q = q_0 q_1 q_2 \ldots q_{N-1}$	конечное множество из N **состояний**
Σ	конечный **входной алфавит**
q_0	**начальное состояние**
F	множество **конечных состояний**
$\delta(q, i) : Q \times \Sigma \rightarrow Q$	**функция перехода** или матрица перехода между состояниями

Алгоритм распознавания для детерминированного КА

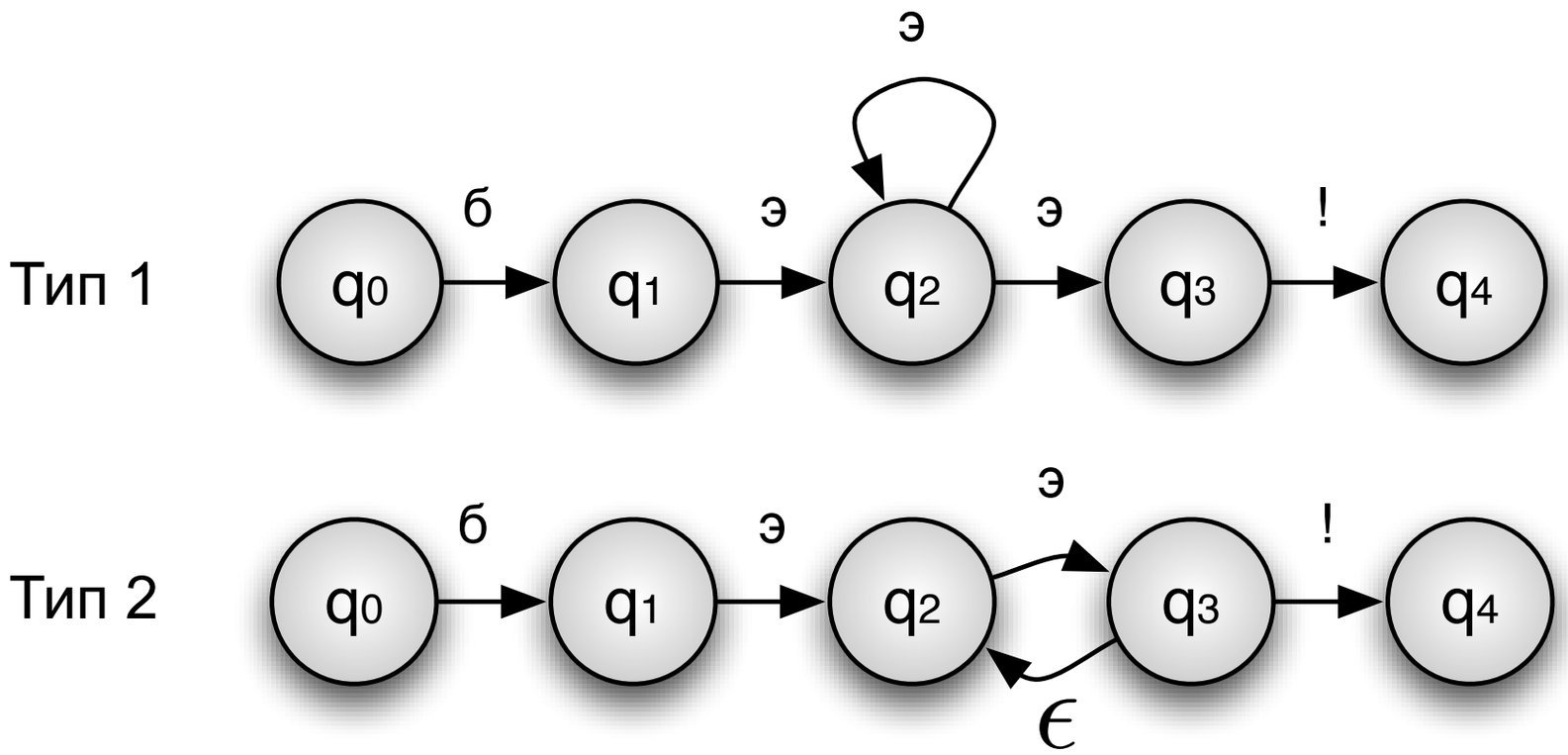
```
#encoding=CP1251
def recognize(tape, machine, acceptStates):
    index = 0 # Beginning of tape
    currentState = 0 # Initial state of machine
    while True:
        if index == len(tape):
            if currentState in acceptStates:
                return True
            else:
                return False
        elif machine[currentState].has_key(tape[index]):
            currentState = machine[currentState][tape[index]]
            index+=1
        else:
            return False

machineSheep = {0:{"б":1}, 1:{"э":2}, 2:{"э":3}, 3:{"э":3,"!":4}, 4:{}}
print recognize("бэээээээ!",machineSheep, [4])
```

	Вход		
	б	э	!
0	1	∅	∅
1	∅	2	∅
2	∅	3	∅
3	∅	3	4
4	∅	∅	∅

Недетерминированные КА

- Обобщение ДКА
- Недетерминизм двух типов



Распознавание для НКА

- Подходы к решению проблемы недетерминизма
  - –Сохранение состояний (backup)
    - поиск в глубину и ширину
  - –Просмотр будущих состояний (look-ahead)
  - –Параллелизм

Состояние	Вход			
	б	э	!	ϵ
0	1	∅	∅	∅
1	∅	2	∅	∅
2	∅	2,3	∅	∅
3	∅	∅	4	∅
4	∅	∅	∅	∅

ДКА и НКА

- ДКА и НКА эквивалентны
- Существует простой алгоритм для преобразования НКА в ДКА
- Идея:
  - взять все параллельные ветки НКА
  - в них взять все состояния, в которых одновременно может находиться НКА
  - объединить их в новое состояние ДКА
- В худшем случае НКА с N состояниями преобразуется в ДКА с 2^N состояниями

Регулярные языки и связь регулярных выражений и конечных автоматов

Регулярные языки

1. $\varnothing$ - регулярный язык
2. $\forall a \in \Sigma \cup \epsilon, \{a\}$ - регулярный язык
3. Для любых регулярных языков L_1 и L_2, такими также являются:
   (a) $L_1 \cdot L_2 = \{xy \mid x \in L_1, y \in L_2\}$, соединение L_1 и L_2
   (b) $L_1 \cup L_2$, объединение или дизъюнкция L_1 и L_2
   (c) L_1^*, замыкание (Клини) языка L_1

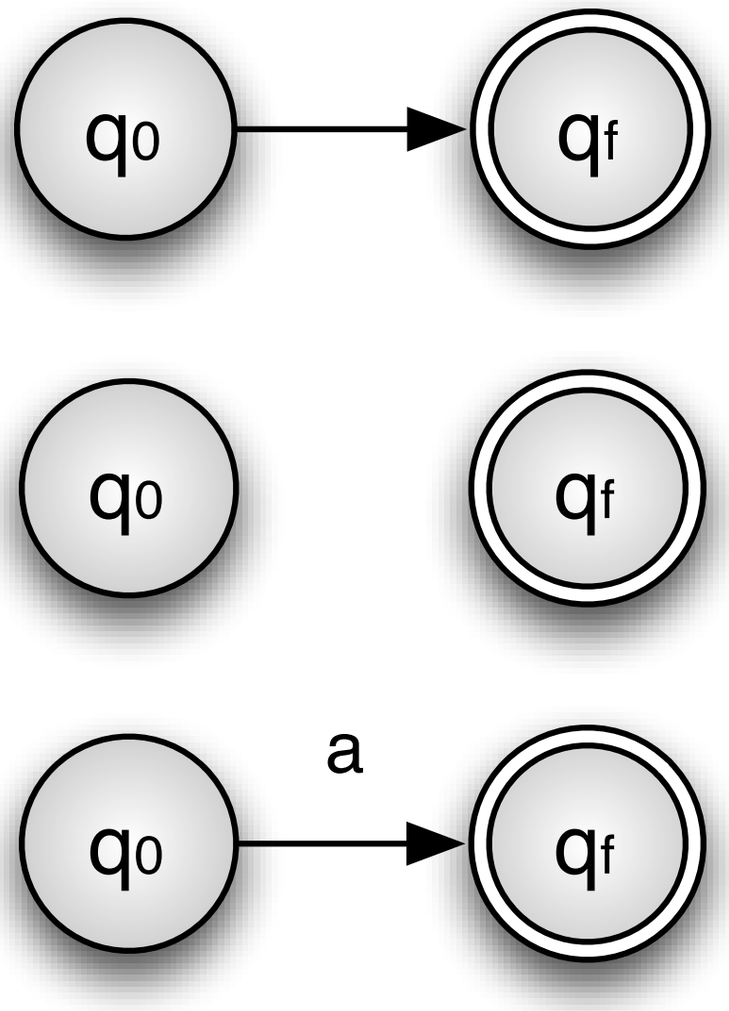
Σ - Алфавит, ϵ - пустая строка

- регулярные языки также замкнуты относительно операций
  - пересечения
  - разности
  - дополнения
  - инверсии

Регулярные языки и регулярные выражения

- Любые операторы регулярных выражений (кроме памяти) можно выразить с помощью трех операций регулярных языков:
  - соединение
  - объединение
  - замыкание Клини
- Пример: счетчики (*, +, {n,m}) являются частным случаем повторения плюс замыкание Клини

Построение автомата для регулярных выражений

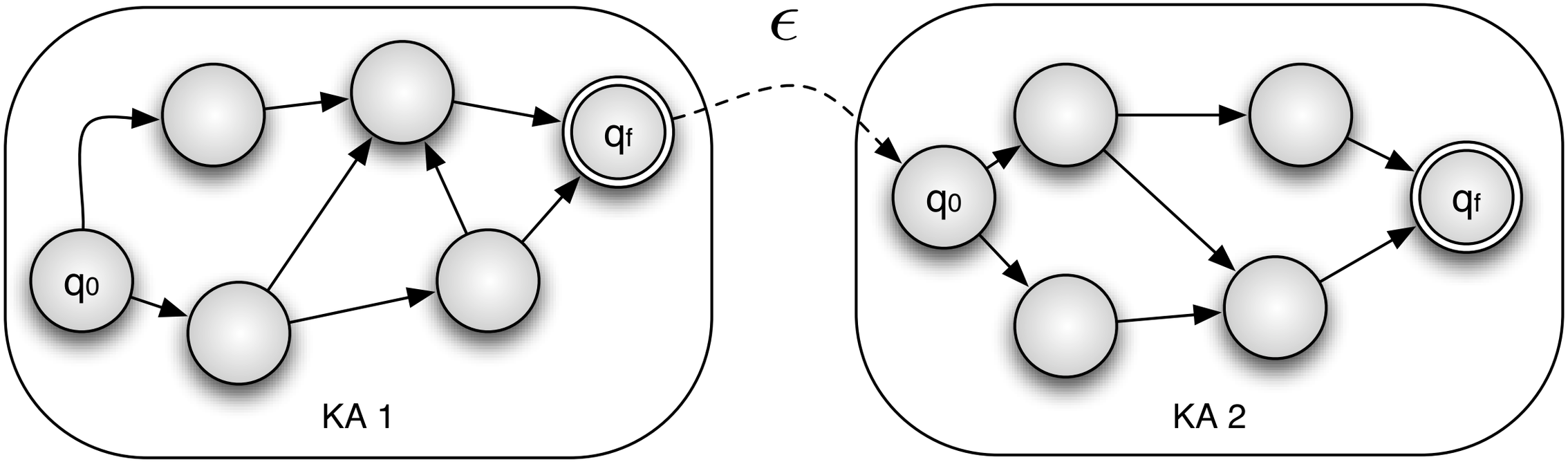


$r = \epsilon$

$r = \varnothing$

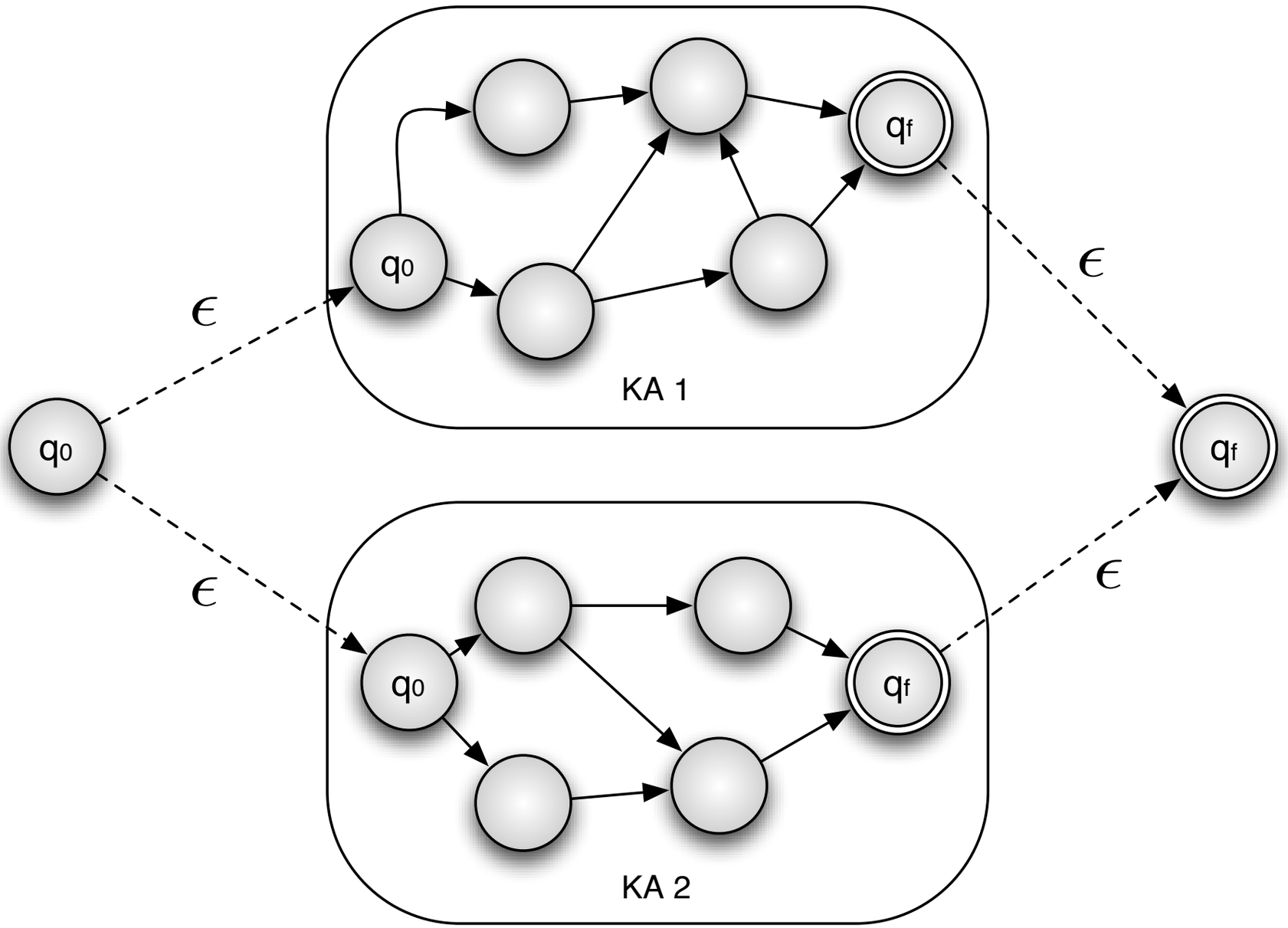
$r = a$

Построение автомата для регулярных выражений



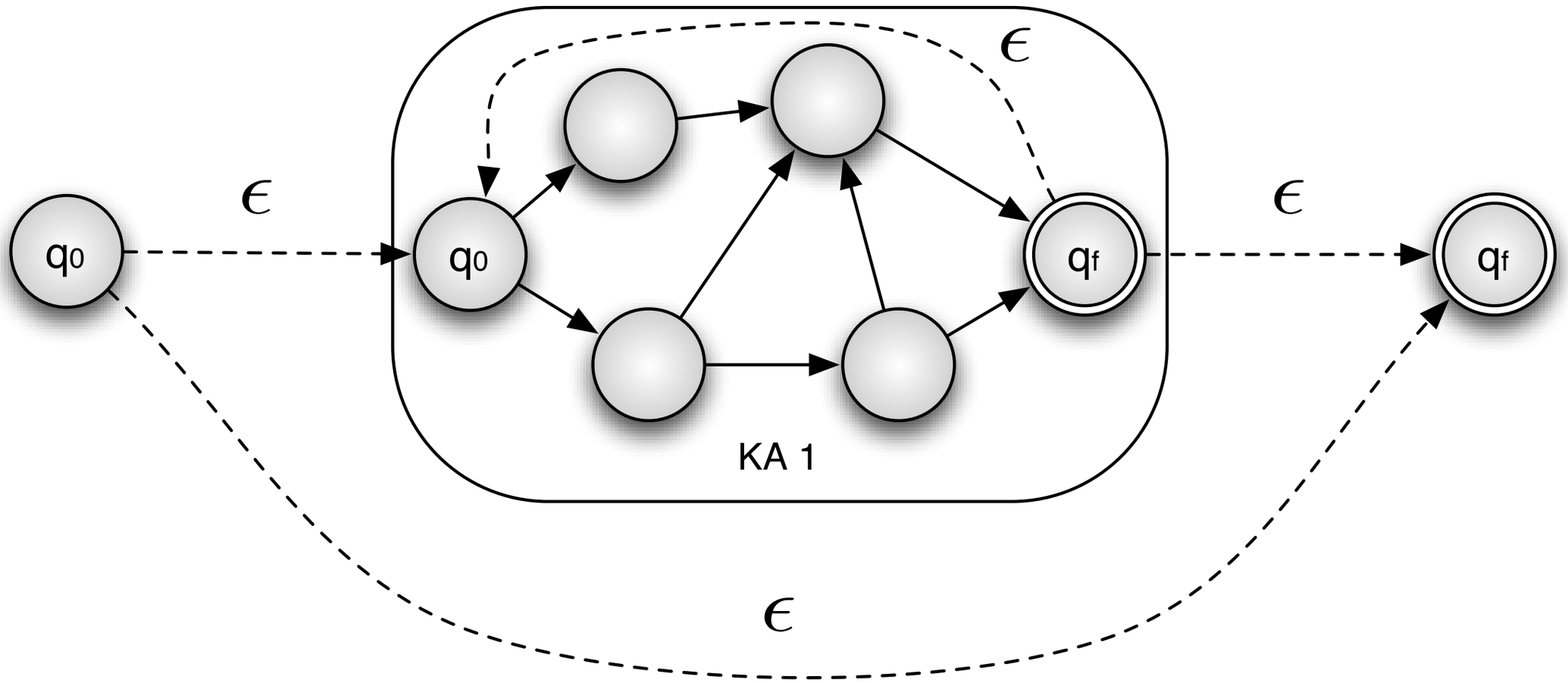
Последовательное соединение двух конечных автоматов

Построение автомата для регулярных выражений



Объединение двух конечных автоматов

Построение автомата для регулярных выражений



Замыкание конечного автомата

Резюме

- Изучены важная фундаментальная концепция **конечных автоматов** и практический инструмент, основанный на ней - **регулярные выражения**
  - регулярные выражения - мощный инструмент для обработки текстов
  - любое регулярное выражение может быть реализовано с помощью КА (кроме памяти)
  - автомат неявно определяет формальный язык
  - для любого НКА существует ДКА

Задания для тренировки

- Написать аналог ELIZA
- Реализовать конечный автомат для распознавания всех русских числительных
- Спроектировать КА для дат: *March 12, the 22nd of November, Christmas*
- Расширить предыдущий автомат относительными датами: *yesterday, tomorrow, a week from tomorrow, the day before yesterday, three weeks from Saturday, next Monday, ...*

Дополнительная литература

- Kleene S.C. **Representation of events in nerve nets and finite automata.** (1951, 1956)

- Rabin M.O. and Scott D. **Finite automata and their decision problems.** (1959)

- Hopcroft J.E. and Ulman J.D. **Introduction to Automata Theory, Languages, and Computations.** (1979)

Следующая лекция

- Языковые модели
- Задача определения частей речи