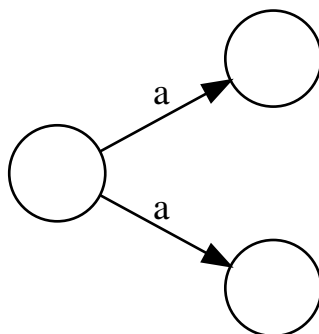


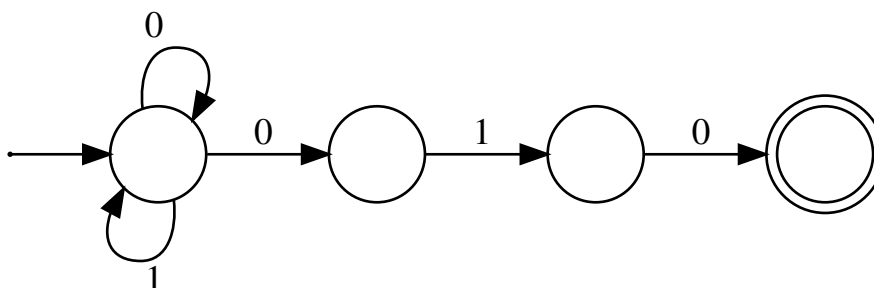
НКА

Уберем ограничение на единственность перехода по символу, тогда получим **недетерминированный КА**:



Кроме того, уберем ограничение на существование перехода, тогда если перехода нет, то слово не допускается в искомый язык. Слово допускается, если существует последовательность недетерминированных выборов, приводящая к допуску.

Пример. Автомат для слов с суффиксом 010:



Алгоритм проверки допуска НКА

$Q = \{1, 2 \dots q\}$ — все вершины

`term` — массив булевых переменных терминальности состояний.

`go[c][i]` — вектор вершин, в которые можно попасть из вершины i по символу c

Построим динамику `can[i][u]` — можно ли за i шагов попасть в состояние u .

`accept(x)`

`can[0][s] = true`

for $i = 0 \dots \text{len}(x) - 1$

$c = x[i]$

for $u = 1 \dots q$

```

        if (can[i][u])
            for v : go[u][c]
                can[i + 1][v] = true
    for u = 1...q
        if (can[len(x)][u] and term[u])
            return true
    return false

```

Применим для автомата слов с суффиксом 010 и слова 01010:

	1	2	3	4
0	+			
1	+	+		
2	+		+	
3	+	+		+
4	+		+	
5	+	+		\oplus

Заметим, что 2 строка = 4 строка и 3 символ равен 5 символу, поэтому 3 строка = 5 строка.

Построим по этой динамике ДКА с булевыми векторами строк в вершинах.

```

ds = 1 << s
for du = 0...2**q-1
    for c - symbol
        dv = 0
        for u = 0...q-1
            if (du & (1 << u)) != 0
                for v : go[u][c]
                    dv |= (1 << v)
        dgo[du][c] = dv
    if (du & term != 0)
        dterm[du] = true

```

Для ускорения можно перебирать не все маски, а только те, которые встретились, т.е. DFS/BFS. Без этого $\mathcal{O}(2^q \cdot \text{poly}(q, \sigma))$, с оптимизацией $\mathcal{O}(\text{ans} \cdot \text{poly}(q, \sigma))$, ans = число состояний ДКА.

Таким образом, мы по любому НКА можем построить ДКА с таким же языком, т.е. выполняются следующие:

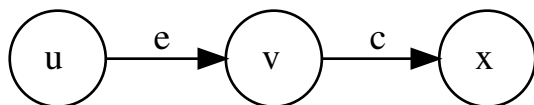
Теорема 1. \forall НКА $A \exists$ ДКА $A_D : L(A) = L(A_D)$

Создадим в НКА ребра, по которым можно перейти, не считав символа, и будем обозначать такое ребро ε . Эта конструкция называется ε -НКА.

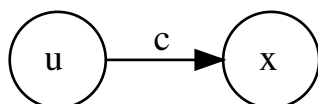
Докажем, что ε -НКА эквивалентен ДКА построением:

1. Построим граф ε -переходов и создадим его транзитивное замыкание. Добавленные замыканием ребра добавим в исходный автомат. Язык автомата от этого не изменился, т.к. переход по новому ребру эквивалентен n переходам по ε -ребрам в исходном графе. Теперь можно не делать два ε -перехода подряд.
2. Если из обычной вершины есть ε -переход в терминальную вершину, то сделаем эту вершину терминальной. Язык автомата от этого преобразования не изменился. Теперь последний переход происходит не по ε .

3. Преобразуем



в

Теперь можно не делать ε -переходы.4. Удалим ε -переходы.**Теорема 2. Клини.**

$$L - \text{регулярный} \Leftrightarrow \exists \text{ ДКА } A : L = L(A)$$

Докажем " \Rightarrow "Докажем " \Leftarrow ".*Доказательство.* $Q := \{1, 2, \dots, q\}$

Построим $\sim q^3$ регулярных выражений, занумеруем их как ξ_{ijk} , $i = 1 \dots q$, $j = 1 \dots q$, $k = 0 \dots q$.

ξ_{ijk} задаёт слова, переводящие автомат из состояния i в j , используя промежуточные состояния с номерами $\leq k$

$\triangleleft k = 0, i = j, \xi_{ii0} = \varepsilon | c_1 | c_2 | \dots$ — из i в i без промежуточных состояний, т.к. состояний ≤ 0 \nexists . c_i — петли в i .

$\xi_{ij0} = c_1 | c_2 | \dots, c_i$ — переход $i \rightarrow j$

$\xi_{ijk+1} = \xi_{ijk} | \xi_{ik+1k} (\xi_{k+1k+1k})^* \xi_{k+1jk}$ — можем либо идти по пути с промежуточными $\leq k$, либо дойти $i \rightarrow k+1$, после чего $n \geq 0$ раз пройти $k+1 \rightarrow k+1$ с промежуточными $\leq k$, после чего дойти $k+1 \rightarrow j$ с промежуточными k (k , а не $k+1$, т.к. номера состояний уникальны, а в $k+1$ мы уже не заходим) \square



Рис. 1:
Ав-
то-
мат
для
 $\{\emptyset\}$



Рис. 2:
Ав-
то-
мат
 $\{\varepsilon\}$



Рис. 3:
Ав-
то-
мат
 $\{c\}$

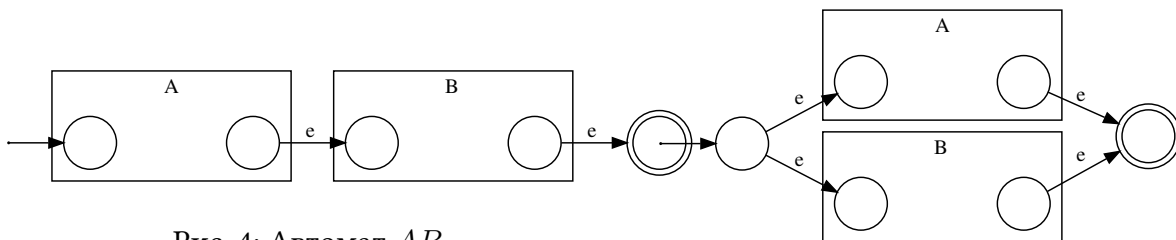


Рис. 4: Автомат AB

Рис. 5: Автомат $A \cup B$



Рис. 6: Автомат A^*