|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № \_\_**1**\_\_**

**Дисциплина Конструирование компиляторов**

|  |  |
| --- | --- |
| **Тема Распознавание цепочек регулярного языка**  **Вариант №6**  **Студент \_Иванов В.А. \_\_\_\_\_\_\_\_**  **Группа \_ИУ7-21М\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель \_Ступников А.А.** |  |

Москва.

2023 г.

**Цель работы**: приобретение практических навыков реализации важнейших элементов лексических анализаторов на примере распознавания цепочек регулярного языка.

Задачи работы:

1. Ознакомиться с основными понятиями и определениями, лежащими в основе построения лексических анализаторов.
2. Прояснить связь между регулярным множеством, регулярным выражением, праволинейным языком, конечноавтоматным языком и недетерминированным конечно-автоматным языком.
3. Разработать, тестировать и отладить программу распознавания цепочек регулярного или праволинейного языка в соответствии с предложенным вариантом грамматики.

**Вариант 6**

Напишите программу, которая в качестве входа принимает произвольное регулярное выражение, и выполняет следующие преобразования:

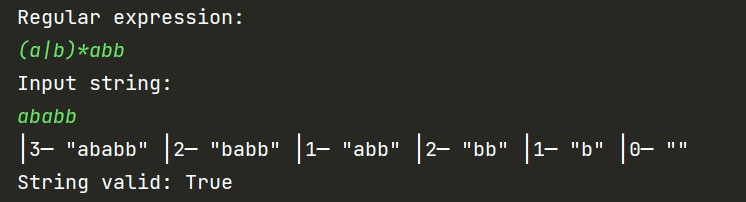
1) Преобразует регулярное выражение непосредственно в ДКА.

2) По ДКА строит эквивалентный ему КА, имеющий наименьшее возможное количество состояний (Воспользоваться алгоритмом Хопкрофта)

3) Моделирует минимальный КА для входной цепочки из терминалов исходной грамматики.

Текст программы и набор тестов приведены в приложении.

Пример работы программы (в режиме отладки):



**Выводы**

В результате выполнения лабораторной работы были изучены основные понятия построения лексических анализаторов, разработана программа распознавания цепочек регулярного языка.

**Контрольные вопросы**

**1) Какие из следующих множеств регулярны? Для тех, которые регулярны, напишите регулярные выражения.**

**2) Найдите праволинейные грамматики для тех множеств из вопроса 1, которые регулярны.**

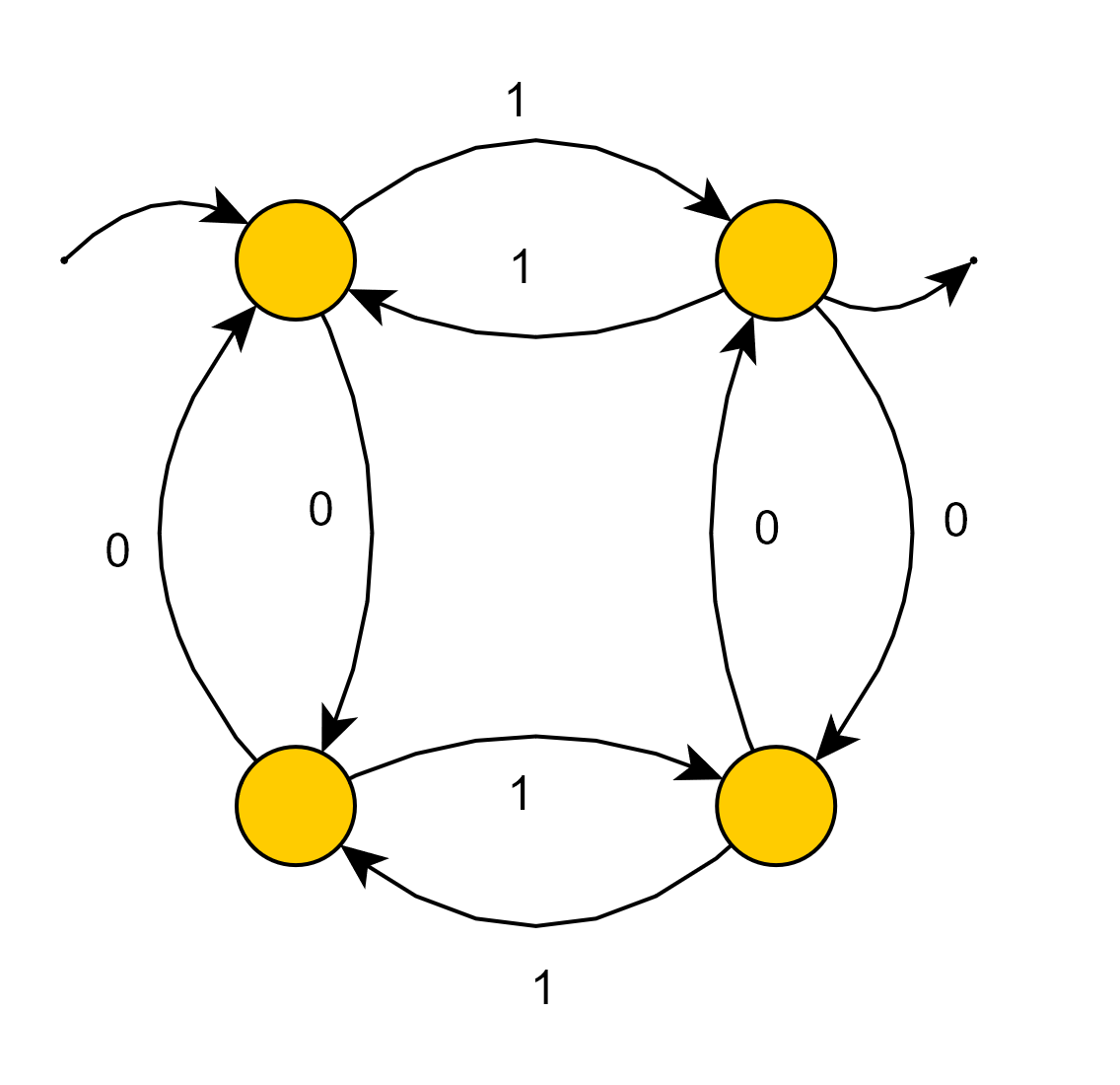
**3) Найдите детерминированные и недетерминированные конечные автоматы для тех множеств из вопроса 1, которые регулярны.**

a. Множество цепочек с равным числом нулей и единиц.

Нерегулярное. Соответствующий автомат будет иметь бесконечность состояний (состояния соответствуют текущей разнице между количеством нулей и единиц).

b. Множество цепочек из {0, 1}\* с четным числом нулей и нечетным числом единиц.

Регулярно. Составим ДКА,



Если существует ДКА, то из него можно получить соответствующее регулярное выражение:

(0(11)\*0)\*(1|0(11)\*10)(0(11)\*0|(1|0(11)\*10)(0(11)\*0)\*(1|0(11)\*10))\*

Праволинейная грамматика:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S→A | A→0B  B→11B  B→C  C→0A  A→D | D→1E  D→0F  F→11F  F→J  J→10E | E→0H  H→11H  H→I  I→0E  E→e | E→1K  E→0L  L→11L  L→M  M→10K | K→0N  N→11N  N→O  O→0K | K→1E  K→0P  P→11P  P→U  U→10E |

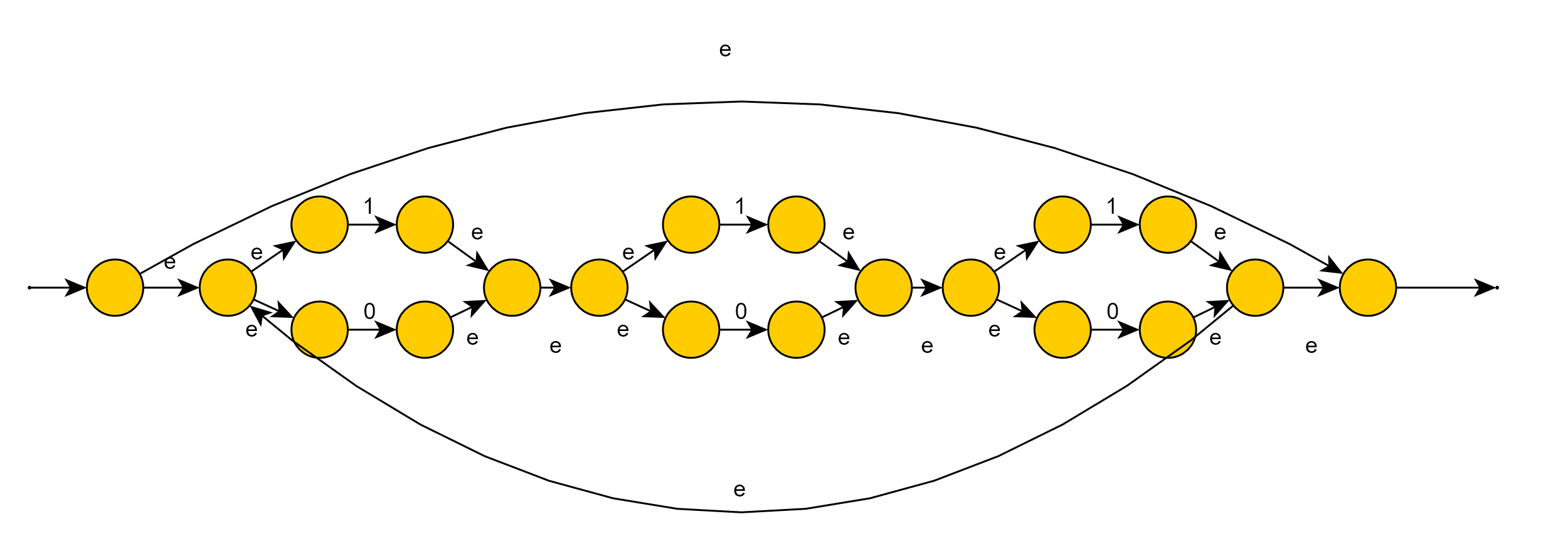
c. Множество цепочек из {0, 1}\*, длины которых делятся на 3.

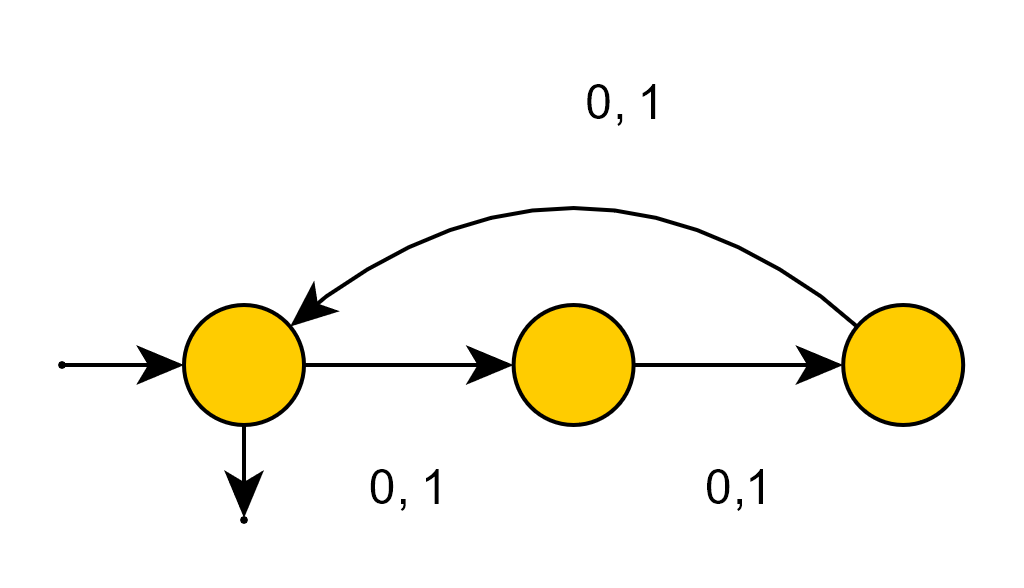
Регулярно.

((0|1)(0|1)(0|1))\*

Праволинейная грамматика:

|  |  |  |  |
| --- | --- | --- | --- |
| S->A | A->0B  A->1B  A->e | B->0C  B->1C | С->0A  C->1A |





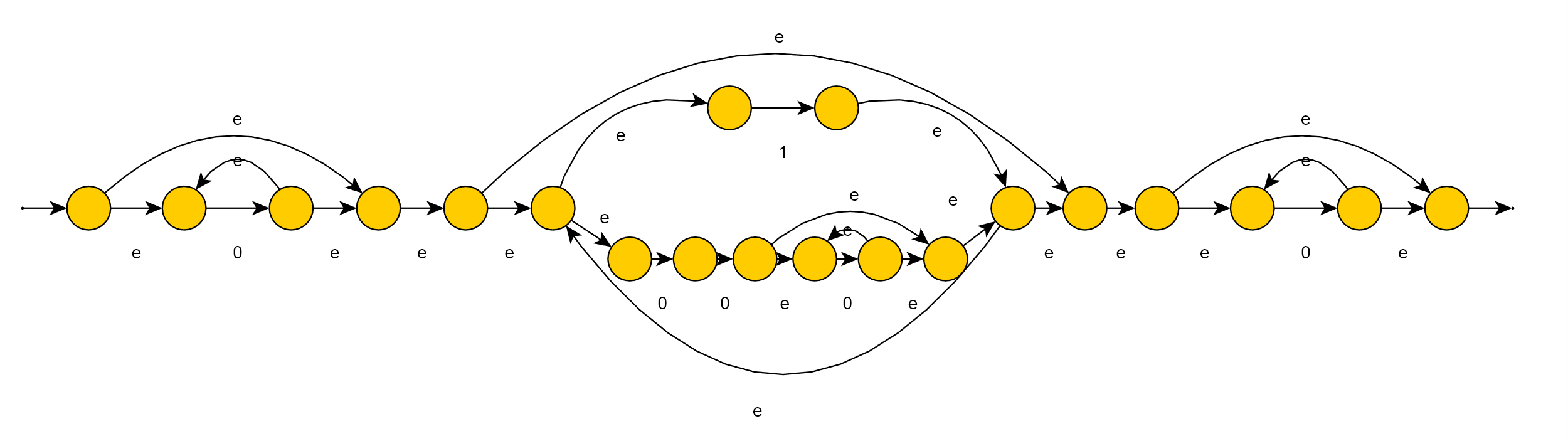
d. Множество цепочек из {0, 1}\*, не содержащих подцепочки 101.

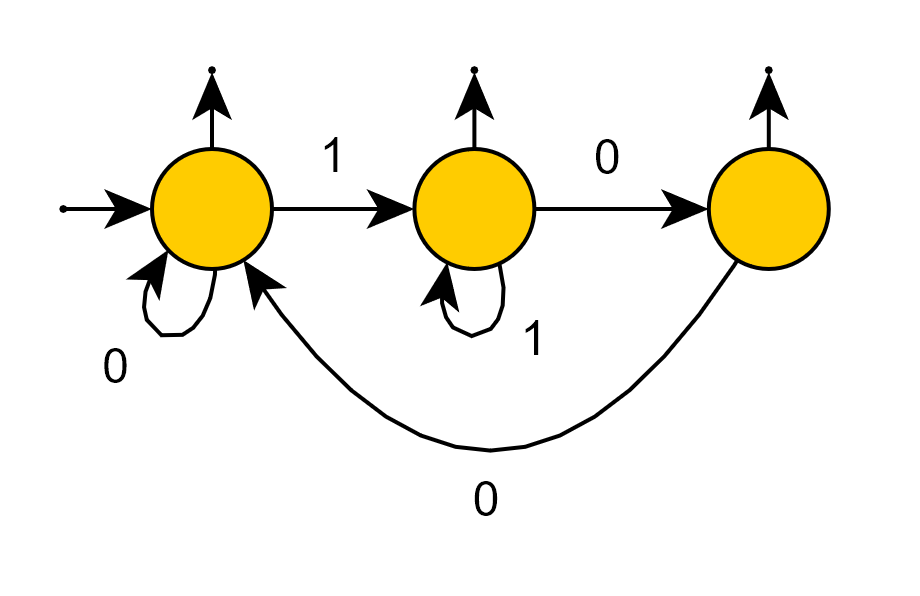
Регулярно.

0\*(1|000\*)\*0\*

Праволинейная грамматика:

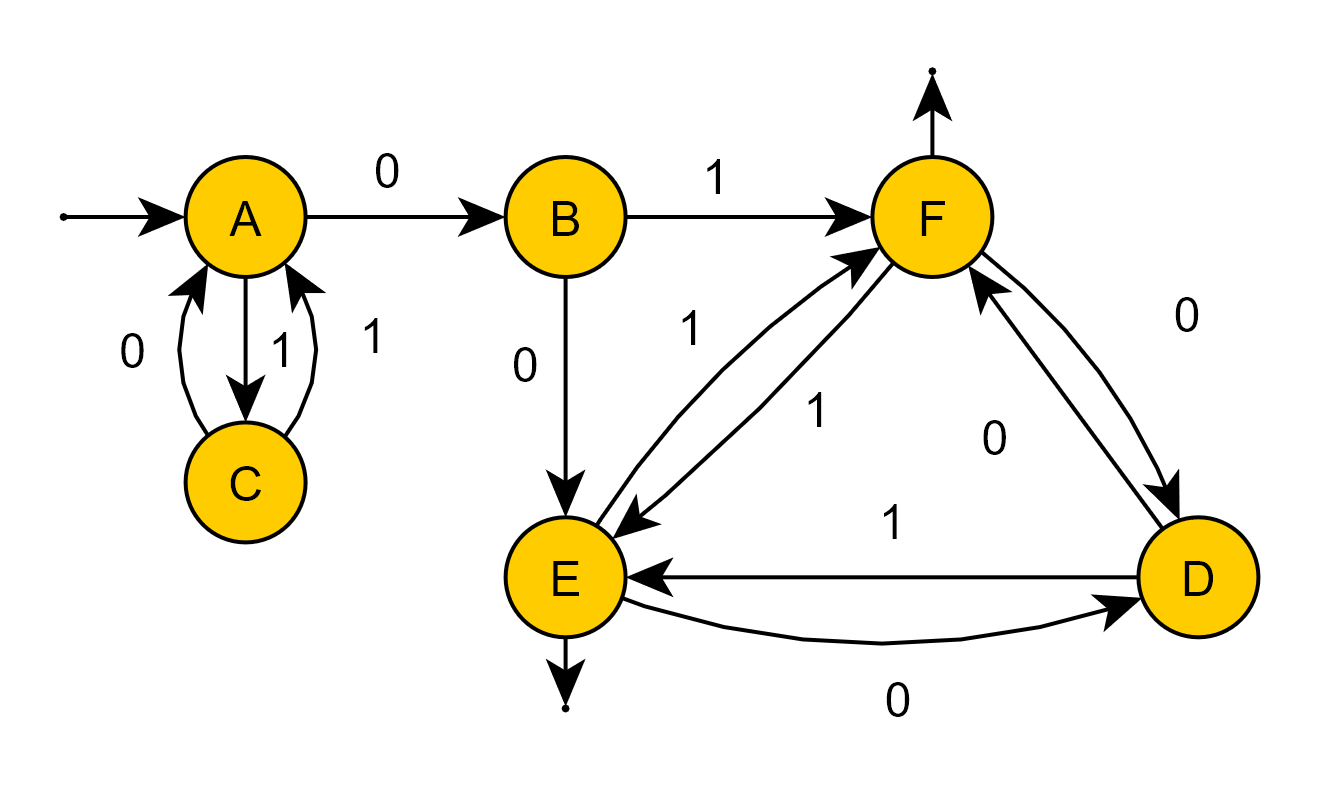
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S->A | A->0A  A->B | B->1B  B->00C  B->D | C->0C  C->B | D->0D  D->e |





**4) Найдите конечный автомат с минимальным числом состояний для языка, определяемого автоматом M = ({A, B, C, D, E}, {0, 1}, d, A, {E, F}), где функция d задается таблицей**





По алгоритму минимизации (алгоритм Хопкрофта)

P = {{A, B, C, D}, {E, F}} – начальное разбиение

Перебираются сплитеры (пара множество состояний - символ).

Сплитер <P0, 1> разбивает P0 на множества {A, C}, {B, D}

P = {{A, C}, {B, D}, {E, F}}

Сплитер <P0, 0> разбивает P0 на множества {A}, {C}

P = {{A}, {C}, {B, D}, {E, F}}

Больше разбиений не происходит. Новый автомат имеет 4 состояния

