Консултация за първо контролно по Дискретни Структури 2

Цветан Ангелов

Март 2017

Съдържание

Част I

Първо контролно

Глава 1

Предговор

Първото контролно по Дискретни Структури 2 включва в себе си задачи от:

- Езици и построяване на автомати по тях;
- Алгоритми за детерминирани и недетерминирани крайни автомати;
- Регулярни изрази;
- Доказване на нерегулярност на език.

В тази част от документа ще разгледаме основни понятия, методи за решаване и примерни задачи, свързани с всеки един от тези типове. Имайте в предвид, че документа по никакъв начин не изчерпва темата, но мисля че е добър начин за опресняване на знанията и подготовка за контролното или писмения изпит.

Глава 2

Езици, понятия и операции свързани с тях

Общи сведения Езиците стоят в основата на втората част от курса по Дискретни структури. Но, за да работим с тях първо трябва да изясним основни понятия:

- **Език** - множество от думи;
- **Дума** - крайна редица от букви (низ);
- **Празна дума** - празен низ, бележим с ε;
- **Буква** - символ;
- **Азбука** - непразно множество от букви.

Казваме, че езика е множество от думи. Следователно е логично и операциите върху множества да са приложими и тук. И това е така с две изключения - декартовото произведение и операцията за множество от подмножества, тъй като техният резултат не е множество от думи. Така остават операциите обединение, сечение и разлика, които са абсолютно еквивалентни на тези за произволно множество. За езиците добавяме и някои нови операции:

- **Конкатенация** - $L_1L_2 = \{w_1w_2 \mid w_1 \in L_1 \text{ и } w_2 \in L_2\}$;
- **n-кратна конкатенация** - $L^n = LLL\ldots L$ - n пъти;
  Забележка: $L^0 = \{\varepsilon\}$.
- **Итерация** - $L^* = L^0 \cup L^1 \cup L^2 \cup \ldots$;

- **Положителна итерация** - $L^+ = L^1 \cup L^2 \cup \ldots$.

  Забележка: $L^+ = LL^* = L^*L$.
  Забележка: $L^* = L^+ \cup \{\varepsilon\} = \{\varepsilon\} \cup L^+$.

Сега нека си припомним курса по алгебра и концепцията за обратна матрица. Ще въведем нещо подобно и за езиците.

$$L^{-1}L = \{\varepsilon\}$$

При така поставената дефиниция имаме "обратен език"само при $L = \{\varepsilon\}$. За това ще разглеждаме следното:

$$x^{-1}y = z \Rightarrow y = xz \text{ - премахваме } x \text{ от началото на } y$$

$$yx^{-1} = z \Rightarrow y = zx \text{ - премахваме } x \text{ от края на } y$$

Нека генерализираме:

$$L^{-1}M = \{w \mid \exists u \in L(uw \in M)\}$$

$$ML^{-1} = \{w \mid \exists u \in L(wu \in M)\}$$

Тоест резултата от тези две операции е език, съдържащ думи, които се получават от дума в M като се премахнат първите / последните няколко символа и тези премахнати символи са дума от L. За простота, когато L съдържа само една дума a ще пишем $a^{-1}M$ и Ma^{-1}.

За финал да споменем някои полезни неща за думи. Нека w е дума. Тогава имаме:

- $N_{<sym>}(w)$ - броят на $<sym>$ в думата w;
- $|w|$ - дължина на думата, брой символи в нея;
- w^R - думата в обратен ред на символите;
- Поддума - поредица от символи в w;
- Префикс - поддума в самото начало на думата (няма символи пред префикса);
- Суфикс - поддума в самият край на думата (няма символи след суфикса).

Задачи Нека видим на практика някои от описаните операции с една примерна задача. Тъй като ще се занимаваме с езици по - обстойно в следващите теми не е необходимо да разглеждаме повече задачи:

Задача 1. (от упражнение при Антон Зиновиев - 23.02.2016 г.) Даден е езикът $M = \{aba, ab, bab, baa\}$. Намерете:

а) $a^{-1}M = \{ba, b\}$ - това са всички думи, които се получават като от дума от M премахнем буквата a от началото;

б) $b^{-1}M = \{ab, aa\}$ - това са всички думи, които се получават като от дума от M премахнем буквата b от началото;

в) $c^{-1}M = \emptyset$ - това са всички думи, които се получават като от дума от M премахнем буквата c от началото;

г) $Ma^{-1} = \{ab, ba\}$ - това са всички думи, които се получават като от дума от M премахнем буквата a от края;

д) $Mb^{-1} = \{a, ba\}$ - това са всички думи, които се получават като от дума от M премахнем буквата b от края;

е) $Mc^{-1} = \emptyset$ - това са всички думи, които се получават като от дума от M премахнем буквата c от края;

ж) $M\{ab\}^{-1} = \{\varepsilon, b\}$ - това са всички думи, които се получават като от дума от M премахнем последователността от букви ab от края (суфикса ab);

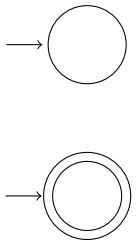
з) $M\{a, b\}^{-1} = \{ab, a, ba\}$ - това са всички думи, които се получават като от дума от M премахнем буквата a или b от края.

Глава 3

Детерминирани крайни автомати и построяването им по даден език

Общи сведения Крайният автомат е устройство с крайна памет. Чрез него ние можем да определяме лесно дали дадена дума е част от езика, определен от автомата. Всеки краен автомат има два основни типа елементи:

- **Състояния** - във всеки един момент автомата се намира в някое от тези състояния. Важно е от кое състояние започва работата на автомата - то се нарича начално състояние и е единствено. Друг важен вид състояния са заключителните, те сигнализират, че ако автомата е в крайно състояние след обработка на целия вход, то входа се приема от автомата. За всяко от тези видове има съответно означение:

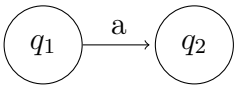


Фигура 3.1: Начално състояние

Фигура 3.2: Обикновено състояние

Фигура 3.3: Заключително състояние

- **Преходи** - те свързват две състояния чрез конкретен символ:



Фигура 3.4: Преход от q_1 към q_2 чрез буквата a

Детерминиран е този автомат, за който няма състояние, от което излизат два прехода с една и съща буква и също така всеки преход е върху реална буква (тоест не може да има преход с празната дума). Нека сега разгледаме една проста схема за построяване на краен детерминиран автомат по зададен език:

1. Започваме със списък от езици с единствен език зададеният от условието;
2. За всеки необработен език L от списъка:
   (а) За всяка буква a от азбуката образуваме езика $a^{-1}L$;
   (б) Ако полученият език не е в списъка, добавяме го.
3. Маркираме L като обработен;
4. Ако има необработени езици в списъка, премини към 2;
5. Всеки от езиците в списъка отговаря на състояние от автомата;
6. Преходите на автомата отговарят на следното условие:
   Нека L_1, L_2 - езици от списъка, a - буква от азбуката. Съществува преход от състоянието L_1 към L_2 с буква $a \Leftrightarrow L_2 = a^{-1}L_1$;
7. Начално състояние е това, отговарящо на зададения в условието език;
8. Крайни състояния са тези, отговарящи на език, съдържащ празната дума.

Задачи Да разгледаме няколко задачи свързани с току що показаната процедура:

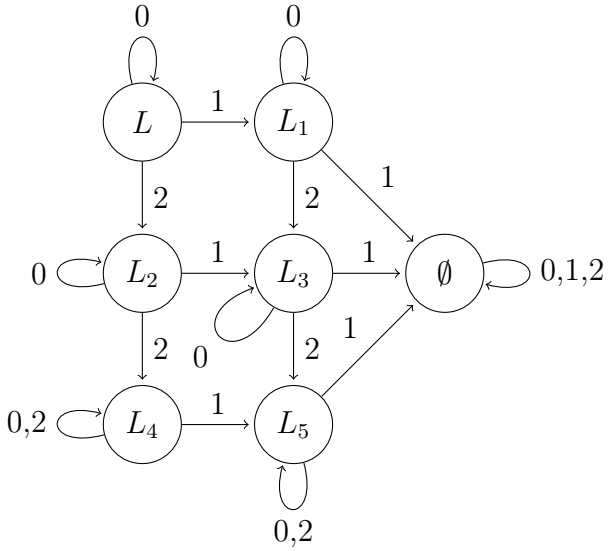
Задача 1. (от упражнение при Антон Зиновиев - 08.03.2016 г.) Намерете краен детерминиран автомат, който отговаря на следния език: $L = \{w \in \{0,1,2\}^* \mid$ в w има най - много една единица (1) и най - малко две двойки $(2)\}$.

Започваме работа по горната процедура. Единствен необработен език е L. Имаме азбука от три символа, следователно получаваме:

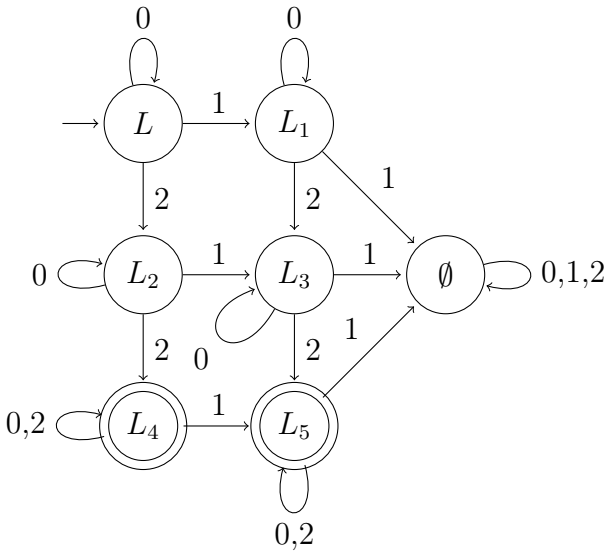
- $0^{-1}L = L$ - това следва от факта, че премахването на началната нула не влияе на критерия за попадане в L;
- $1^{-1}L = \{w \in \{0,1,2\}^* \mid$ в w няма нито една единица (1) и най - малко две двойки $(2)\} = L_1$;
- $2^{-1}L = \{w \in \{0,1,2\}^* \mid$ в w има най - много една единица (1) и най - малко една двойка $(2)\} = L_2$;
- $0^{-1}L_1 = L_1$;
- $1^{-1}L_1 = \emptyset$;
- $2^{-1}L_1 = \{w \in \{0,1,2\}^* \mid$ в w няма нито една единица (1) и най - малко една двойка $(2)\} = L_3$;
- $0^{-1}L_2 = L_2$;
- $1^{-1}L_2 = L_3$;
- $2^{-1}L_2 = \{w \in \{0,1,2\}^* \mid$ в w има най - много една единица $(1)\} = L_4$;
- $0^{-1}L_3 = L_3$;
- $1^{-1}L_3 = \emptyset$;
- $2^{-1}L_3 = \{w \in \{0,1,2\}^* \mid$ в w няма нито една единица $(1)\} = L_5$;
- $0^{-1}L_4 = L_4$;
- $1^{-1}L_4 = \{w \in \{0,1,2\}^* \mid$ в w няма нито една единица $(1)\} = L_5$;
- $2^{-1}L_4 = L_4$;

- $0^{-1}L_5 = L_5$;
- $1^{-1}L_5 = \emptyset$;
- $2^{-1}L_5 = L_5$;
- $0^{-1}\emptyset = \emptyset$;
- $1^{-1}\emptyset = \emptyset$;
- $2^{-1}\emptyset = \emptyset$.

Така в списъка вече няма необработени езици. От всяко едно от тези равенства знаем какви преходи има в автомата. Така получаваме следното:



Сега е време да определим и начално и крайни състояния. Тъй като първоначалният език е L, то началното състояние е L. За крайните състояния проверяваме всеки от езиците дали включва празната дума. Очевидно такива са само L_4 и L_5. Следователно крайното решение е:



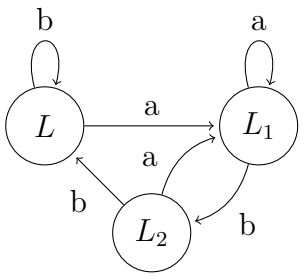
Задача 2. (от контролно - 11.04.2016 г.) Постройте краен детерминиран автомат с език:

$$L = \{w \in \{a, b\}^* \mid w \text{ завършва с } ab\}$$

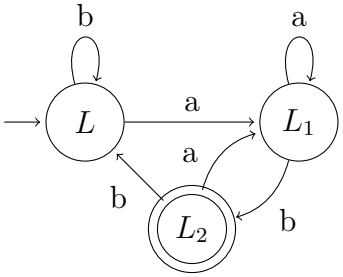
Ще решим задачата отново по горната процедура:

- $a^{-1}L = \{w \in \{a, b\}^* \mid w \text{ завършва с } ab\} \cup \{b\} = L \cup \{b\} = L_1$;
- $b^{-1}L = \{w \in \{a, b\}^* \mid w \text{ завършва с } ab\} = L$;
- $a^{-1}L_1 = a^{-1}\{b\} \cup a^{-1}L = \emptyset \cup L_1 = L_1$;
- $b^{-1}L_1 = b^{-1}\{b\} \cup b^{-1}L = \{\varepsilon\} \cup L = L_2$;
- $a^{-1}L_2 = a^{-1}\{\varepsilon\} \cup a^{-1}L = \emptyset \cup L_1 = L_1$;
- $b^{-1}L_2 = b^{-1}\{\varepsilon\} \cup b^{-1}L = \emptyset \cup L = L$.

Така получаваме следното:



От тези получени езици само един съдържа празната дума (L_2). За това състоянието L_2 е единственото заключително, и отново L е начално. Така крайният резултат е:



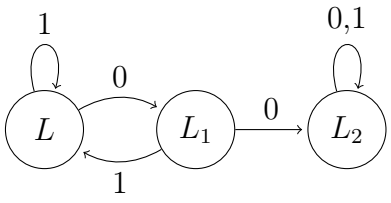
Задача 3. (от контролно - 11.04.2016 г.) Постройте краен детерминиран автомат с език:

$$L = \{w \in \{0,1\}^* \mid w \text{ съдържа като поддума } 00\}$$

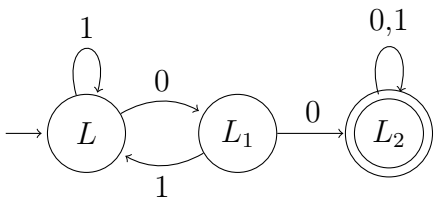
Ще решим задачата отново по горната процедура:

- $0^{-1}L = \{w \in \{0,1\}^* \mid w \text{ съдържа като поддума } 00\} \cup \{w \in \{0,1\}^* \mid w \text{ започва с } 0\} = L \cup \{w \in \{0,1\}^* \mid w \text{ започва с } 0\} = L_1$;
- $1^{-1}L = \{w \in \{0,1\}^* \mid w \text{ съдържа като поддума } 00\} = L$;
- $0^{-1}L_1 = 0^{-1}L \cup 0^{-1}\{w \in \{0,1\}^* \mid w \text{ започва с } 0\} = L_1 \cup \{w \in \{0,1\}^*\} = \{w \in \{0,1\}^*\} = L_2$;
- $1^{-1}L_1 = 1^{-1}L \cup 1^{-1}\{w \in \{0,1\}^* \mid w \text{ започва с } 0\} = L \cup \emptyset = L$;
- $0^{-1}L_2 = 0^{-1}\{w \in \{0,1\}^*\} = L_2$;
- $1^{-1}L_2 = 1^{-1}\{w \in \{0,1\}^*\} = L_2$.

Така получаваме следното:



От тези получени езици само един съдържа празната дума (L_2). За това състоянието L_2 е единственото заключително, и отново L е начално. Така крайният резултат е:



Задача 4. (от контролно - 24.04.2014 г.) Постройте краен детерминиран автомат с език:

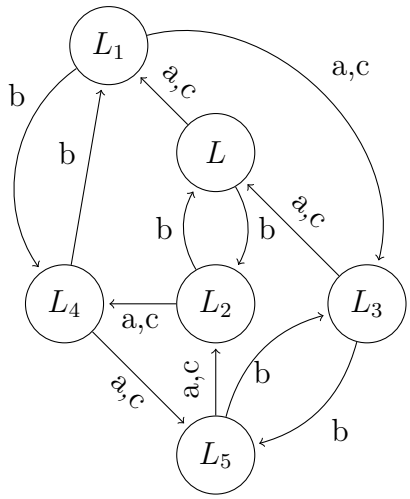
$$L = \{w \in \{a,b,c\}^* \mid (N_a(w) + N_c(w)) \mod 3 = 1 \text{ и } N_b(w) \mod 2 = 0\}$$

Ще решим задачата отново по горната процедура:

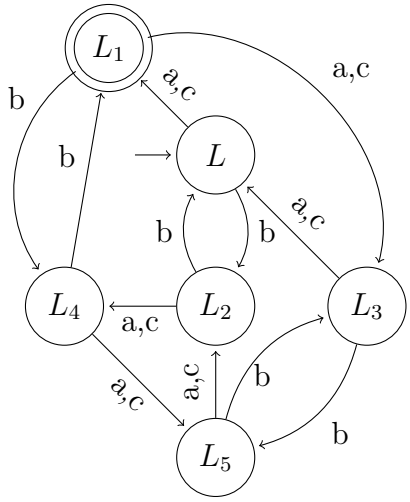
- $a^{-1}L = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 0 \text{ и } N_b(w) \mod 2 = 0\} = L_1$;
- $b^{-1}L = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 1 \text{ и } N_b(w) \mod 2 = 1\} = L_2$;
- $c^{-1}L = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 0 \text{ и } N_b(w) \mod 2 = 0\} = L_1$;
- $a^{-1}L_1 = \{w \in \{a,b,c\}^* \mid (N_a(w) + N_c(w)) \mod 3 = 2 \text{ и } N_b(w) \mod 2 = 0\} = L_3$;
- $b^{-1}L_1 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 0 \text{ и } N_b(w) \mod 2 = 1\} = L_4$;
- $c^{-1}L_1 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 2 \text{ и } N_b(w) \mod 2 = 0\} = L_3$;
- $a^{-1}L_2 = \{w \in \{a,b,c\}^* \mid (N_a(w) + N_c(w)) \mod 3 = 0 \text{ и } N_b(w) \mod 2 = 1\} = L_4$;
- $b^{-1}L_2 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 1 \text{ и } N_b(w) \mod 2 = 0\} = L$;
- $c^{-1}L_2 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 0 \text{ и } N_b(w) \mod 2 = 1\} = L_4$;
- $a^{-1}L_3 = \{w \in \{a,b,c\}^* \mid (N_a(w) + N_c(w)) \mod 3 = 1 \text{ и } N_b(w) \mod 2 = 0\} = L$;
- $b^{-1}L_3 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 2 \text{ и } N_b(w) \mod 2 = 1\} = L_5$;

- $c^{-1}L_3 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 1 \text{ и } N_b(w) \mod 2 = 0\} = L$;
- $a^{-1}L_4 = \{w \in \{a,b,c\}^* \mid (N_a(w) + N_c(w)) \mod 3 = 2 \text{ и } N_b(w) \mod 2 = 1\} = L_5$;
- $b^{-1}L_4 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 0 \text{ и } N_b(w) \mod 2 = 0\} = L_1$;
- $c^{-1}L_4 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 2 \text{ и } N_b(w) \mod 2 = 1\} = L_5$;
- $a^{-1}L_5 = \{w \in \{a,b,c\}^* \mid (N_a(w) + N_c(w)) \mod 3 = 1 \text{ и } N_b(w) \mod 2 = 1\} = L_2$;
- $b^{-1}L_5 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 2 \text{ и } N_b(w) \mod 2 = 0\} = L_3$;
- $c^{-1}L_5 = \{w \in \{a,b,c\}^* \mid (N_a(w)+N_c(w)) \mod 3 = 1 \text{ и } N_b(w) \mod 2 = 1\} = L_2$.

Така получаваме следното:



От тези получени езици само един съдържа празната дума (L_1). За това състоянието L_1 е единственото заключително, и отново L е начално. Така крайният резултат е:


L_1
a,c
b
a,c
b
L
b
b
a,c
L_4
a,c
L_2
L_3
a,c
a,c
b
b
L_5

Глава 4

Алгоритми за детерминирани и недетерминирани крайни автомати

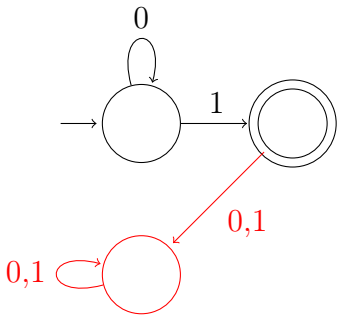
Общи сведения Първо нека уточним какво означава недетерминиран автомат. В миналата глава уточнихме какво е детерминиран автомат. Казваме, че детерминирания е частен случай на недетерминирания при посочените в предната глава условия.

Друго важно определение е **тотален автомат** - такъв автомат, за който от всяко състояние излиза поне един преход с всяка буква от азбуката.

Ето и алгоритмите, които ще използваме:

- За детерминирани:
  - **Тотализация - резултатът е детерминиран тотален краен автомат:**
    1. Визуализираме автомата;
    2. Добавяме ново некрайно състояние;
    3. За всяко състояние от първоначалния автомат добавяме преход към създаденото в 2. състояние с всяка буква от азбуката, с която няма преход от въпросното състояние;
    4. Правим примка на състоянието от 2. с всяка буква от азбуката.

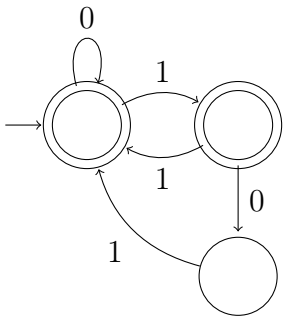
    Пример (червеното се получава в резултат от алгоритъма, азбуката се състои от 0 и 1):

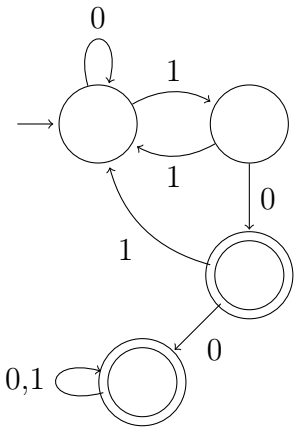


– **Допълнение - резултатът е детерминиран тотален краен автомат:**

1. Тотализираме автомата;
2. За всяко състояние, ако е крайно го правим некрайно, а ако е некрайно го правим крайно.

Пример (първият автомат е входния, втория е получен в резултат от алгоритъма):

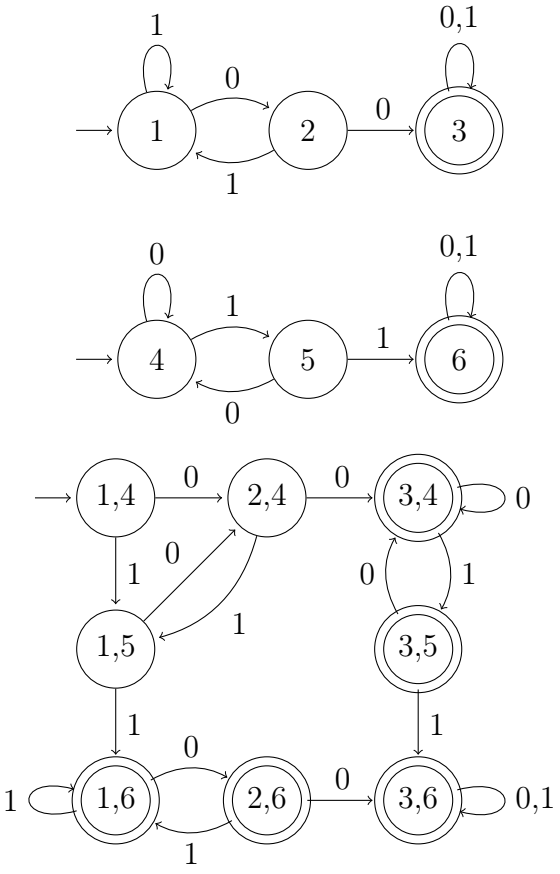




– **Обединение (I) - входът ни е от два детерминирани автомата, резултатът е детерминиран автомат:**

1. Резултатният автомат в началото има единствено състояние, което е началното му състояние и се получава като двойка от началните състояния на входните автомати;
2. За всяко състояние S, от което не излизат преходи, създаваме преход с всяка буква l от азбуката. Всеки от тези преходи води до състояние, което е двойка от състояния на входните автомати, всяко от които се достига с l за една стъпка от някое от състоянията на S в съответния му входен автомат;
3. Ако операцията в 2. доведе до създаване на ново състояние, премини към 2;
4. За всяко от състоянията на резултатния автомат, ако в двойката, която го съставя поне едно от състоянията е крайно във входните автомати, то това състояние е крайно в резултатния автомат.

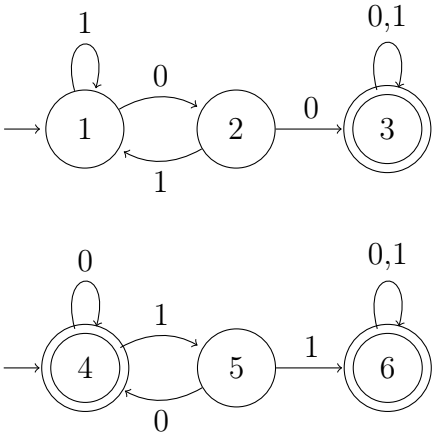
Пример (първите два автомата са входните, третия е получен в резултат от алгоритъма):

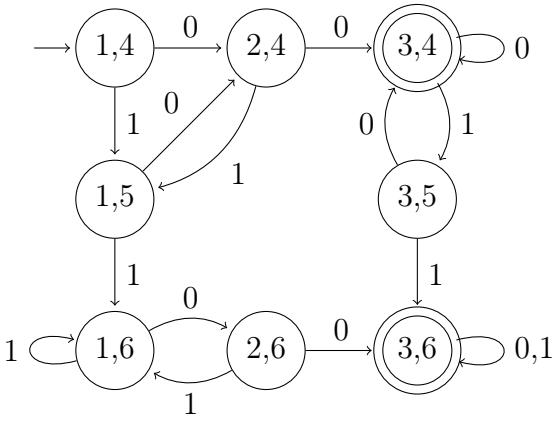


– **Сечение - входът ни е от два детерминирани автомата, резултатът е детерминиран автомат:**

1. Резултатният автомат в началото има единствено състояние, което е началното му състояние и се получава като двойка от началните състояния на входните автомати;
2. За всяко състояние S, от което не излизат преходи, създаваме преход с всяка буква l от азбуката. Всеки от тези преходи води до състояние, което е двойка от състояния на входните автомати, всяко от които се достига с l за една стъпка от някое от състоянията на S в съответния му входен автомат;
3. Ако операцията в 2. доведе до създаване на ново състояние, премини към 2;
4. За всяко от състоянията на резултатния автомат, ако в двойката, която го съставя и двете състояния са крайни в съответния им входен автомат, то това състояние е крайно в резултатния автомат.

Пример (първите два автомата са входните, третия е получен в резултат от алгоритъма):

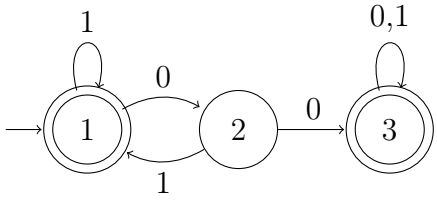


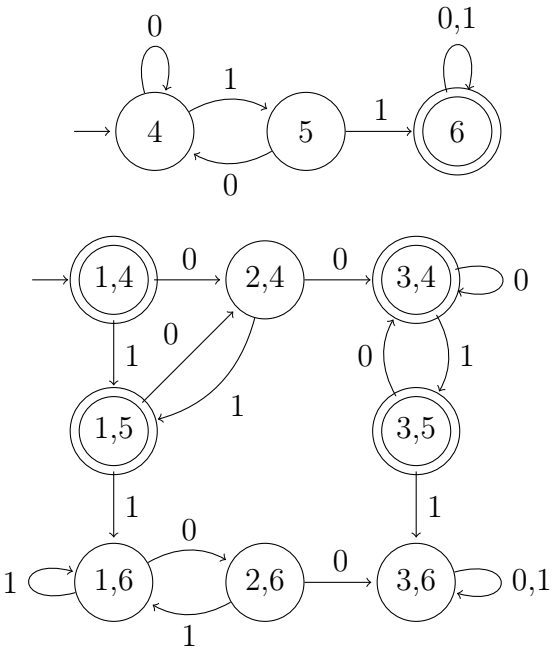


– **Разлика - входа ни е от два детерминирани автомата M1 и M2 и търсим $M1 \backslash M2$, който е детерминиран автомат:**

1. Резултатният автомат в началото има единствено състояние, което е началното му състояние и се получава като двойка от началните състояния на входните автомати;
2. За всяко състояние S, от което не излизат преходи, създаваме преход с всяка буква l от азбуката. Всеки от тези преходи води до състояние, което е двойка от състояния на входните автомати, всяко от които се достига с l за една стъпка от някое от състоянията на S в съответния му входен автомат;
3. Ако операцията в 2. доведе до създаване на ново състояние, премини към 2;
4. За всяко от състоянията на резултатния автомат, ако в двойката, която го съставя първото е крайно в M1 и второто не е крайно в M2, то това състояние е крайно в резултатния автомат.

Пример (първите два автомата са входните, третия е получен в резултат от алгоритъма):





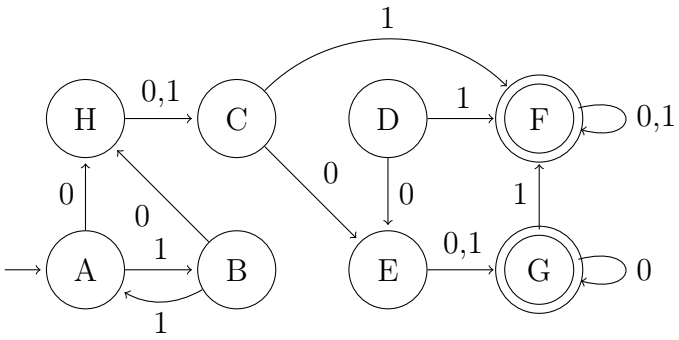
– **Минимизация - входът е краен детерминиран автомат, резултатът е краен детерминиран автомат.** Минимален е такъв краен детерминиран автомат, който измежду всички КДА, разпознаващи същия език има възможно най - малко състояния. Минималният автомат е единствен с точност изоморфизъм.

Алгоритъм на Муур / Moore reduction procedure:

1. Създаваме две множества - първото съдържа началните състояния на входния автомат, а второто - крайните;
2. Правим таблица с резултатите от преходите от всяко от състоянията на автомата за всеки възможен вход. Под резултат тук разбираме не състоянието, до което води прехода, а множеството, в което е въпросното състояние (от текущия набор множества);
3. За всяко от текущите множества, групираме състоянията му на база поведение (две състояния имат еднакво поведение ако за всеки вход стигат до еднакви множества - гледаме от таблицата, получена в 2.). Ако множеството съдържа повече от една група. Премахваме множеството, а групите стават нови самостоятелни множества;
4. Ако на стъпка 3. са образувани нови множества, премини към 2.;

5. Всяко от наличните множества се обединяват в състояние, което е крайно ако множеството е съставено от крайни състояния (крайни във входния автомат). Начално състояние е това, което съдържа началното на входния автомат.
6. Преходите се определят от стойностите получени в стъпка 2. на последната итерация. Тоест гледаме за кое да е от състоянията съставящи дадено множество и гледаме с всевъзможните входове към кое множество се преминава и се пренася в контекста на новия автомат.

Пример (първият автомат е входния):



На първия етап имаме следното:

$$P_1 = \{A, B, C, D, E, H\}$$

$$P_2 = \{F, G\}$$

Сега навлизаме в итерационната част на алгоритъма:

Състояние	Преход с 0	Преход с 1
A	P_1	P_1
B	P_1	P_1
C	P_1	P_2
D	P_1	P_2
E	P_2	P_2
F	P_2	P_2
G	P_2	P_2
H	P_1	P_1

Сега очевидно всички състояния в P_2 имат еднакво поведение. За това то няма да се разпада. От друга страна в P_1 има три различни модела на поведение. Така се получават следните множества:

$$P_2 = \{F, G\}$$

$$P_3 = \{A, B, H\}$$

$$P_4 = \{C, D\}$$

$$P_5 = \{E\}$$

Отново изчисляваме таблицата:

Състояние	Преход с 0	Преход с 1
A	P_3	P_3
B	P_3	P_3
C	P_5	P_2
D	P_5	P_2
E	P_2	P_2
F	P_2	P_2
G	P_2	P_2
H	P_4	P_4

Виждаме, че само в P_3 имат поне две различни поведения, така разбиваме и получаваме:

$$P_2 = \{F, G\}$$

$$P_4 = \{C, D\}$$

$$P_5 = \{E\}$$

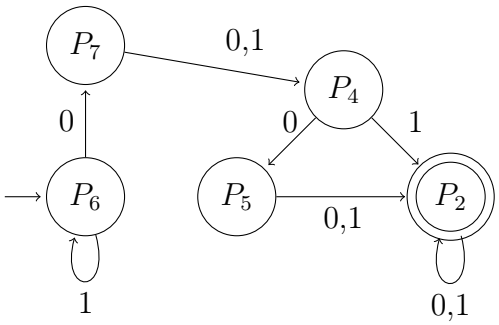
$$P_6 = \{A, B\}$$

$$P_7 = \{H\}$$

Отново изчисляваме таблицата:

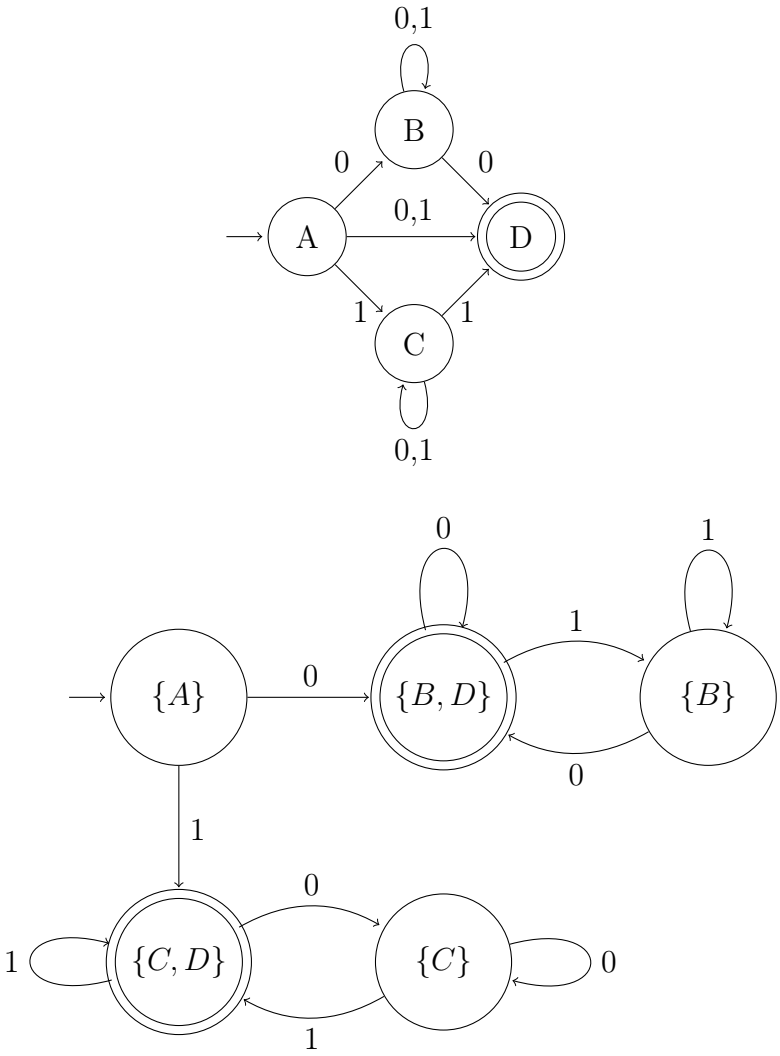
Състояние	Преход с 0	Преход с 1
A	P_7	P_6
B	P_7	P_6
C	P_5	P_2
D	P_5	P_2
E	P_2	P_2
F	P_2	P_2
G	P_2	P_2
H	P_4	P_4

От тази таблица се вижда, че няма да има повече разбивания, така получаваме следния резултатен автомат:



- За недетерминирани (валидни и за детерминирани):
  - **Тотализация - резултатът е тотален краен автомат:** Процедурата е същата като тотализацията при детерминираните;
  - **Детерминизация (Subset Construction Algorithm) - входът е произволен краен автомат, резултатът е детерминиран краен автомат:**
    1. Началното състояние на резултатния автомат отговаря на множество от тези състояния на входния автомат достижими от началното му чрез празната дума (това включва и самото начално състояние);
    2. За новополученото състояние с множество M на резултатния автомат, с всяка буква l от азбуката създаваме преход от това новополучено състояние до друго, чието множество е F. F се получава по следния начин:
       (а) F първоначално се състои от всяко състояние на входния автомат достижимо с l за една стъпка от кое да е състояние в M;
       (б) Към F добавяме всяко състояние на входния автомат достижимо с празната дума от кое да е състояние от F.
    3. За всяко новополучено в 2. състояние стъпката се повтаря. Това продължава докато няма състояние в изходния автомат, което да не е обработено чрез 2;
    4. Заключителни са тези състояния на резултатния автомат, чиито множества съдържат поне едно състояние, което е крайно във входния.

    Пример (първият автомат е входния, втория е получен в резултат от алгоритъма, азбуката е 0 и 1):



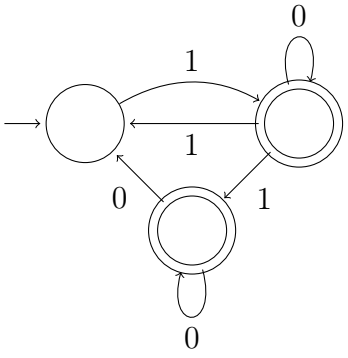
ЗАБЕЛЕЖКА: За следващите операции има и друг вариант да се реализират - чрез преходи с празната дума. Те няма да бъдат разглеждани тук, но са приложими в програмирането. Ярък пример за това е конструкцията на Томпсън, за съставяне на автомат по регулярен израз.

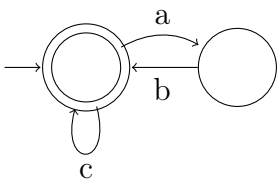
– **Обединение (II) - входът ни е от два произволни крайни автомата, резултатът е краен автомат:**

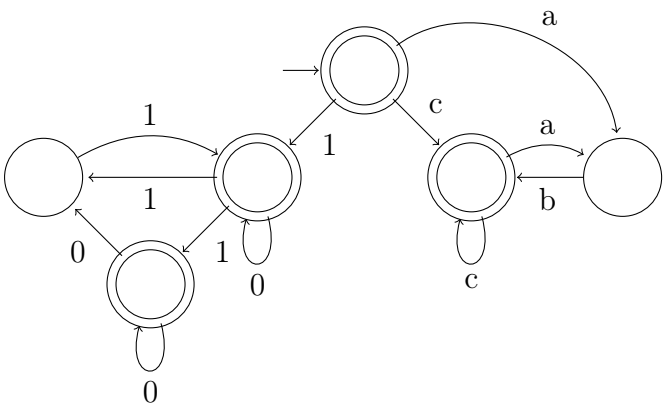
1. Изобразяваме автоматите един до друг;
2. Добавяме ново начално състояние, което е крайно ако поне едно от старите начални състояние е крайно. В противен случай е некрайно;

3. Правим новосъздаденото състояние да върши работата на предишните начални.

Пример (първите два автомата са входните, третия е получен в резултат от алгоритъма):





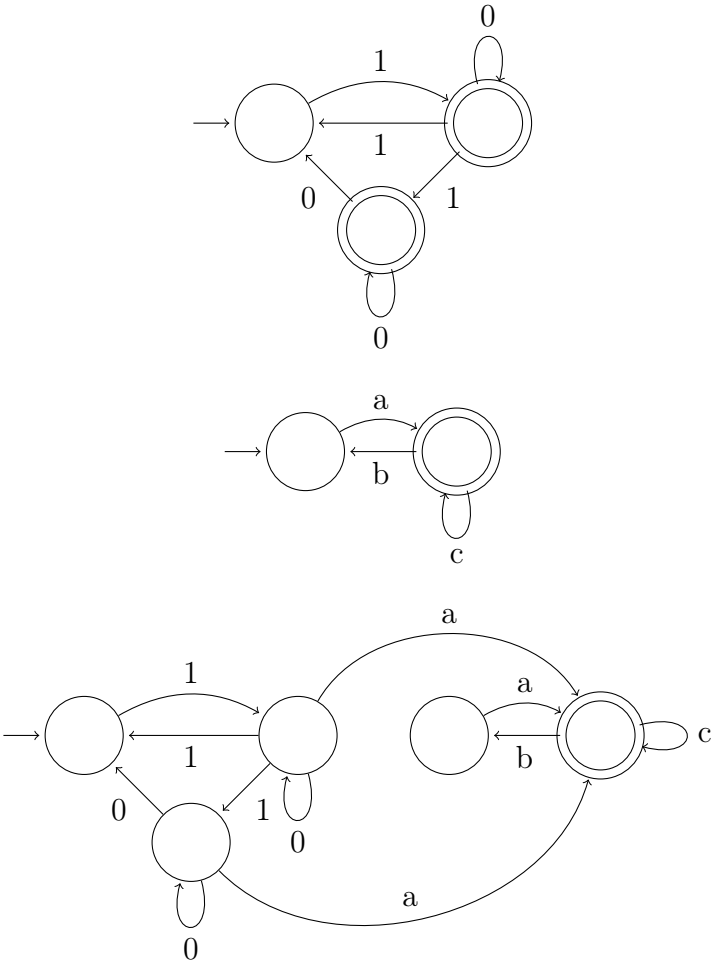


– **Конкатенация - входът ни е от два произволни крайни автомата, резултатът е краен автомат:**

1. Изобразяваме автоматите;
2. Началното състояние на първия автомат е новото начално състояние;
3. Правим всяко крайно състояние на първия да върши работата на началното състояние на втория;

4. Крайните състояния на първият остават крайни единствено ако началното на втория е крайно.

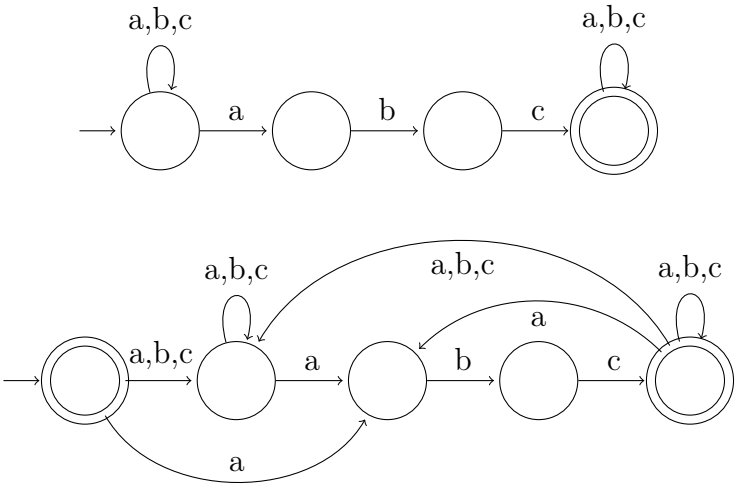
Пример (първите два автомата са входните, третия е получен в резултат от алгоритъма):



– **Итерация (Звезда на Клини) - входът е произволен краен автомат, резултатът е детерминиран краен автомат:**

1. Изобразяваме автомата;
2. Добавяме ново начално състояние, което е крайно;
3. Правим всяко крайно състояние да върши работата на старото начално.

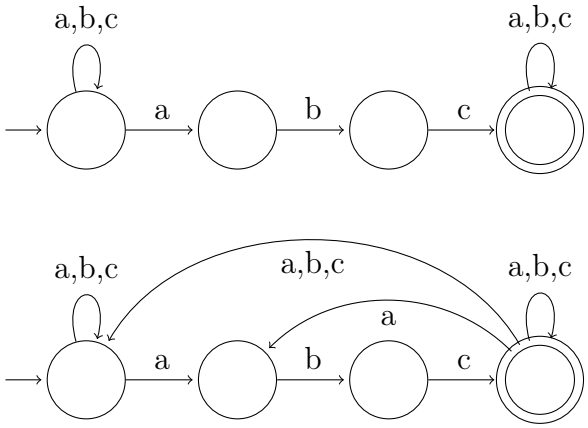
Пример (първият автомат е входния, втория е получен в резултат от алгоритъма):



– **Положителна итерация - входът е произволен краен автомат, резултатът е детерминиран краен автомат:**

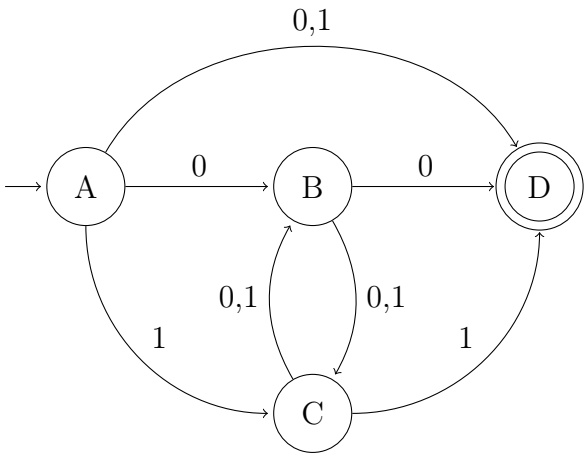
1. Изобразяваме автомата;
2. Правим всяко крайно състояние да върши работата на старото начално.

Пример (първият автомат е входния, втория е получен в резултат от алгоритъма):



Задачи Да разгледаме няколко задачи свързани с разгледаните алгоритми:

Задача 1. (от контролно - 24.04.2014 г.) Намерете краен детерминиран автомат, еквивалентен на:



Прилагаме алгоритъма за детерминизация:

1. Началното състояние на новият автомат е съставено единствено от A - началното на входния;

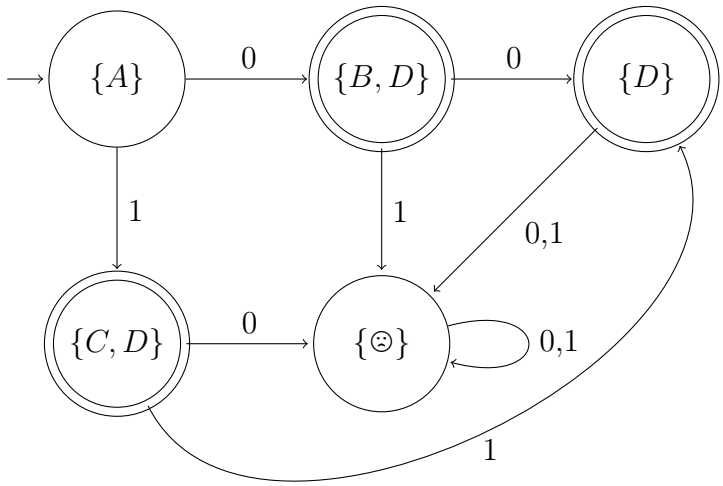
2. От това състояние с 0 се стига до B и D, а с 1 - C и D. Така получаваме две нови състояния;

3. От B и D се стига с 0 съответно до D и "небитието а с 1 - в "небитието"и при двата случая. От C и D се стига с 1 съответно до D и "небитието а с 1 - в "небитието"и при двата случая;

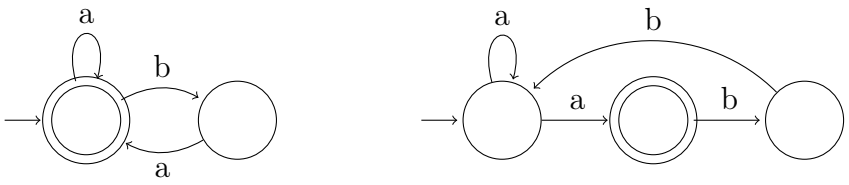
4. От D се стига и с двата символа до "небитието";

5. От "небитието"се стига и с двата символа до "небитието".

Така КДА-то изглежда така:

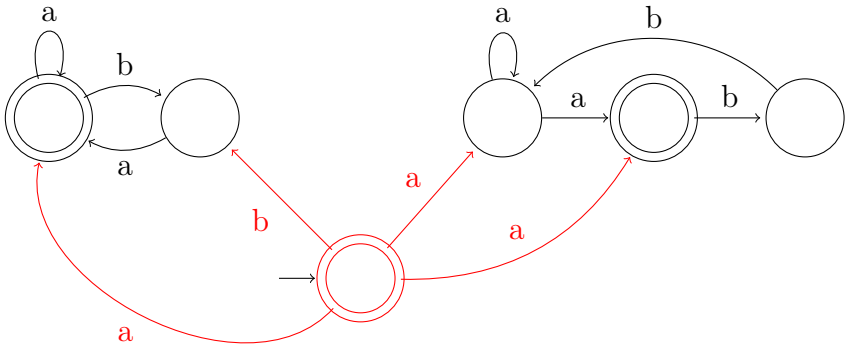


Задача 2. (от контролно - 16.04.2015 г.) Нека L' и L'' са езиците, разпознавани от автоматите

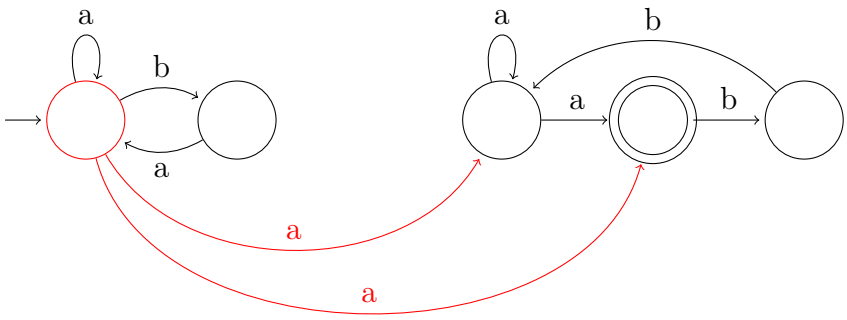


Да се построят крайни автомати за $L' \cup L''$ и $L'.L''$.

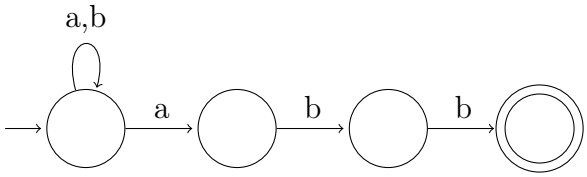
Да започнем с обединението. Директно прилагаме алгоритъма за обединение и получаваме следното (създаденото в резултат от алгоритъма е в червено):



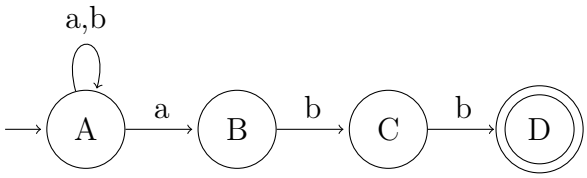
Сега за конкатенацията. Отново прилагаме директно разгледания по - горе алгоритъм за съответната операция:



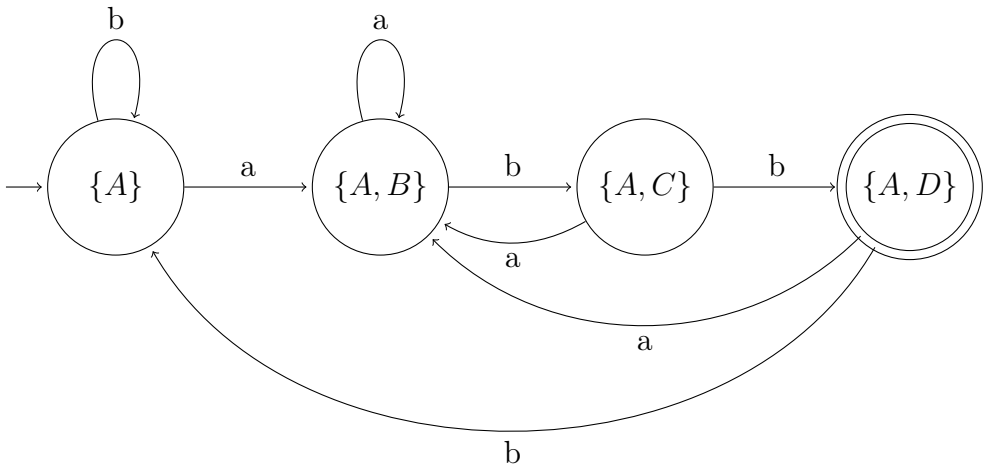
Задача 3. (от контролно - 16.04.2015 г.) Да се детерминира автоматът



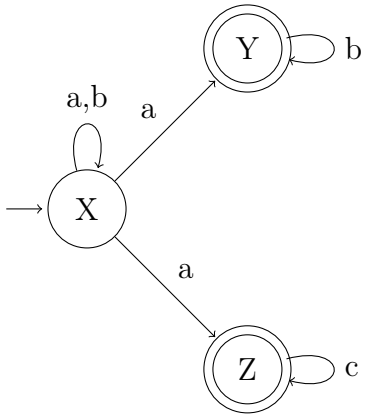
Тук за улеснение ще дадем име на всяко от състоянията:



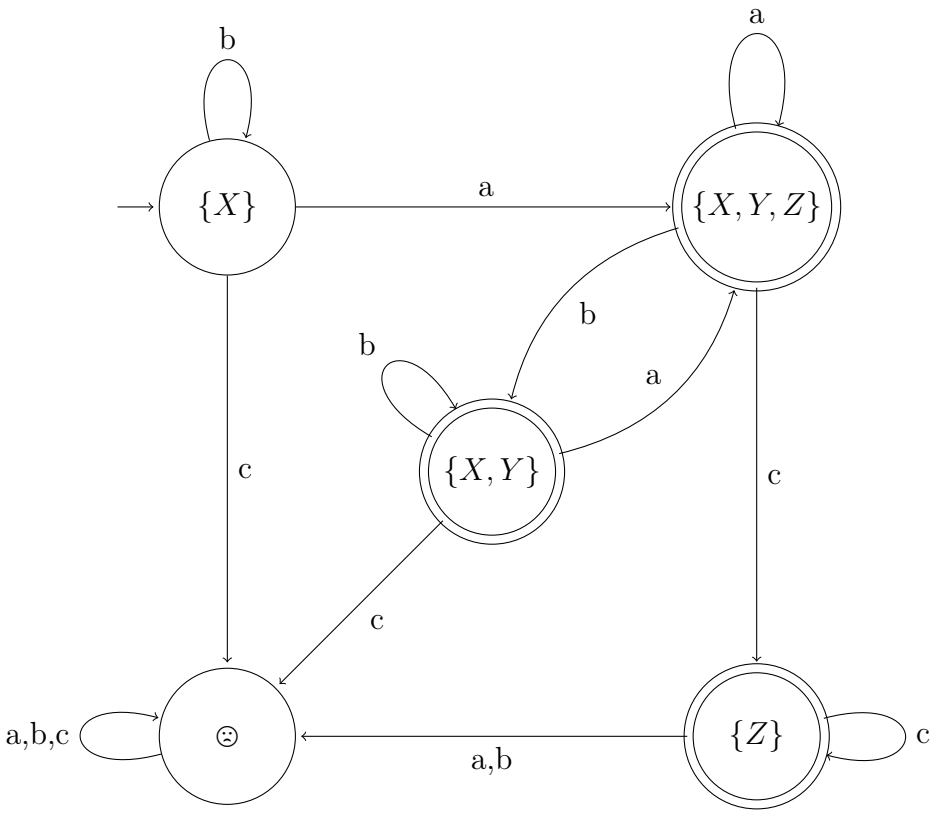
Сега прилагаме алгоритъма и получаваме:



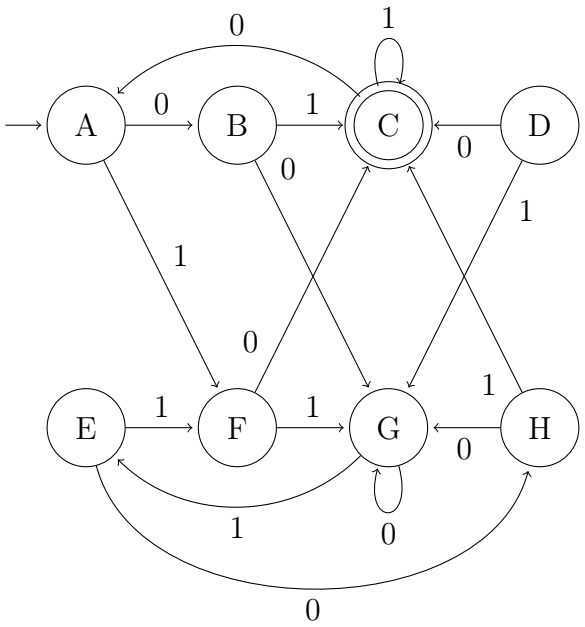
Задача 4. (от пример на Clemson University) Да се детерминира автоматът



Сега прилагаме алгоритъма и получаваме:



Задача 5. (от пример на University of California, Riverside) Минимизирайте автомата:



На първия етап имаме следното:

$$P_1 = \{A, B, D, E, F, G, H\}$$

$$P_2 = \{C\}$$

Сега навлизаме в итерационната част на алгоритъма:

Състояние	Преход с 0	Преход с 1
A	P_1	P_1
B	P_1	P_2
C	P_1	P_2
D	P_2	P_1
E	P_1	P_1
F	P_2	P_1
G	P_1	P_1
H	P_1	P_2

Сега очевидно всички състояния в P_2 имат еднакво поведение (все пак има само едно). За това то няма да се разпада. От друга страна в P_1 има три различни модела на поведение. Така се получават следните множества:

$$P_2 = \{C\}$$

$$P_3 = \{A, E, G\}$$

$$P_4 = \{B, H\}$$

$$P_5 = \{D, F\}$$

Отново изчисляваме таблицата:

Състояние	Преход с 0	Преход с 1
A	P_4	P_5
B	P_3	P_2
C	P_3	P_2
D	P_2	P_3
E	P_4	P_5
F	P_2	P_3
G	P_3	P_3
H	P_3	P_2

Виждаме, че само в P_3 имат поне две различни поведения, така разбиваме и получаваме:

$$P_2 = \{C\}$$

$$P_4 = \{B, H\}$$

$$P_5 = \{D, F\}$$

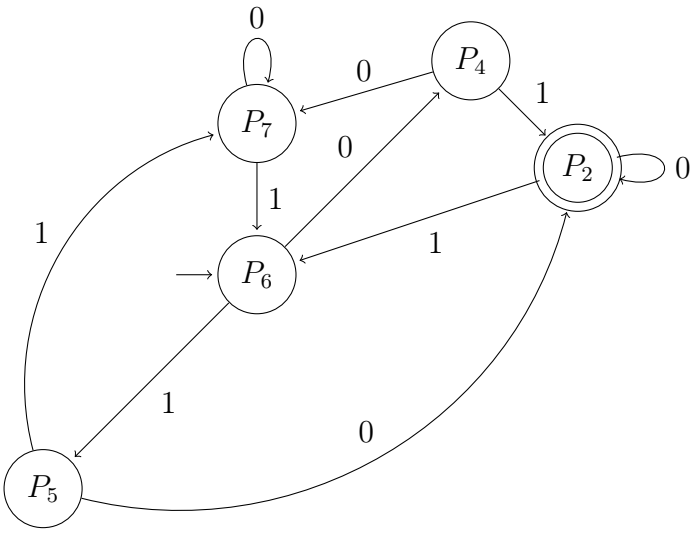
$$P_6 = \{A, E\}$$

$$P_7 = \{G\}$$

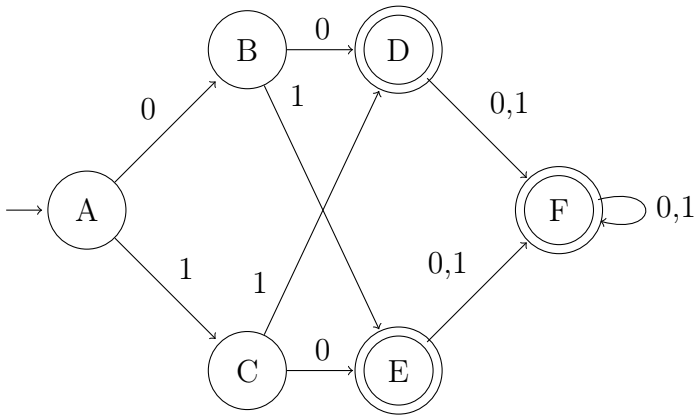
Отново изчисляваме таблицата:

Състояние	Преход с 0	Преход с 1
A	P_4	P_5
B	P_7	P_2
C	P_6	P_2
D	P_2	P_7
E	P_4	P_5
F	P_2	P_7
G	P_7	P_6
H	P_7	P_2

От тази таблица се вижда, че няма да има повече разбивания, така получаваме следния резултатен автомат:



Задача 6. (от пример на Cornell University) Минимизирайте автомата:



На първия етап имаме следното:

$$P_1 = \{A, B, C\}$$

$$P_2 = \{D, E, F\}$$

Сега навлизаме в итерационната част на алгоритъма:

Състояние	Преход с 0	Преход с 1
A	P_1	P_1
B	P_2	P_2
C	P_2	P_2
D	P_2	P_2
E	P_2	P_2
F	P_2	P_2

Сега очевидно всички състояния в P_2 имат еднакво поведение. За това то няма да се разпада. От друга страна в P_1 има два различни модела на поведение. Така се получават следните множества:

$$P_2 = \{D, E, F\}$$

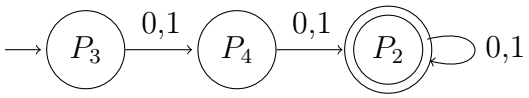
$$P_3 = \{A\}$$

$$P_4 = \{B, C\}$$

Отново изчисляваме таблицата:

Състояние	Преход с 0	Преход с 1
A	P_4	P_4
B	P_2	P_2
C	P_2	P_2
D	P_2	P_2
E	P_2	P_2
F	P_2	P_2

От тази таблица се вижда, че няма да има повече разбивания, така получаваме следния резултатен автомат:



Глава 5

Регулярни изрази

Общи сведения Регулярните изрази са мощно средство за създаване на шаблони, които после могат да се използват за операции върху текстове. Те всъщност се свеждат до автомати тъй като определят регулярен език. Сега първо ще разгледаме основата на синтаксиса на регулярните изрази, след което ще разгледаме и как да получим автомат.
Синтаксис:

Математически синтаксис	Значение
a	буквата "a"
$\{a, b, c, d, e, 5\}$	Някоя от изброените букви
$\{a, b, \ldots, e, 5\}$	Еквивалентно на предходния ред
Σ	Коя да е буква от азбуката
ε	Празната дума
$R_1 R_2$	Конкатенация
$R_1 \cup R_2$	Обединение
R^*	Итерация
R^+	Положителна итерация

Важно е да се знае какъв приоритет имат тези операции. От висок към нисък се подреждат така: двете итерации, конкатенация, обединение. Този приоритет се изменя чрез използване на кръгли скоби.

Нека сега да видим как от един език ще образуваме съответния му регулярен израз:

$$L_1 = \{w \in \{0,1\}^* \mid w \text{ съдържа поне две поддуми } 001\}$$

Това означава, че ние ще искаме да имаме дума, която на всяка позиция има коя да е буква освен на 6 позиции, групирани в две групи от три

последователни символа. За тези групи искаме всяка да бъде 001. Тоест получаваме следният регулярен израз:

$$\Sigma^*001\Sigma^*001\Sigma^*$$

Като правим регулярен израз по език трябва много да внимаваме за частните случаи. Нека видим следния пример:

$$L_2 = \{w \in \{0,1\}^* \mid w \text{ съдържа поне две поддуми } 010\}$$

На пръв поглед същото като при L_1. Но това не е точно така. Тук е възможен случай, в който двете поддуми се препокриват частично. Тоест може нашите две поддуми да са прилепени и крайната 0 на първата да се препокрие с началната на втората - 01010. Тоест ние трябва да хванем и този случай:

$$\Sigma^*010\Sigma^*010\Sigma^* \cup \Sigma^*01010\Sigma^*$$

Това си личи, че има доста излишества, така че ще направим малки промени. Виждаме, че в началото имаме пълно съвпадение. Такова имаме и в края. Така може да разделим изразите на три части свързани с конкатенация. Първата и третата се препокриват напълно, така че остава за общата заявка да направим обединение само на двете средни части. Така получаваме:

$$\Sigma^*010(\Sigma^*0 \cup \varepsilon)10\Sigma^*$$

Сега нека хванем този израз и да видим как може да получим автомат:
Може да забележим, че изразът ни може да се разбие на някакви прости изрази, за които много лесно може да оформим автомат. Тях после може да обработим с алгоритмите от Глава 4. В случая ще разбием на 4 израза:

а) Σ^*010;

б) Σ^*0;

в) ε;

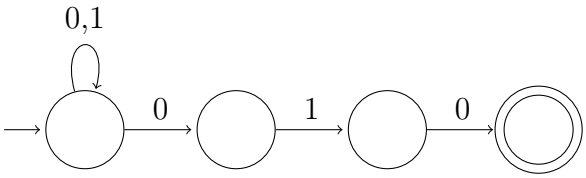
г) $10\Sigma^*$.

Така нашият израз вече изглежда така:

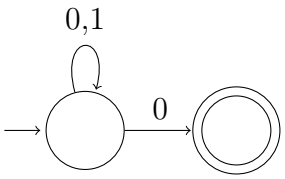
$$\text{а}(\text{б} \cup \text{в})\text{г}$$

Нека сега направим всеки от четирите автомата:

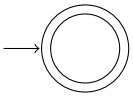
а) Автоматът а



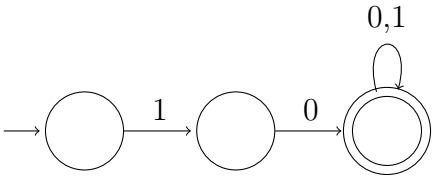
б) Автоматът б



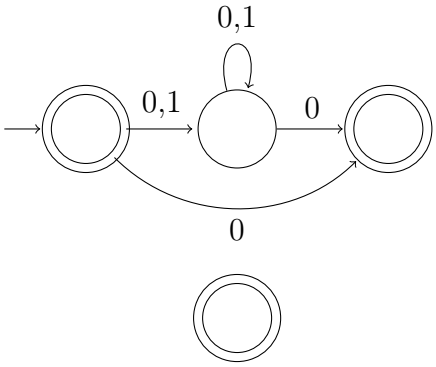
в) Автоматът в



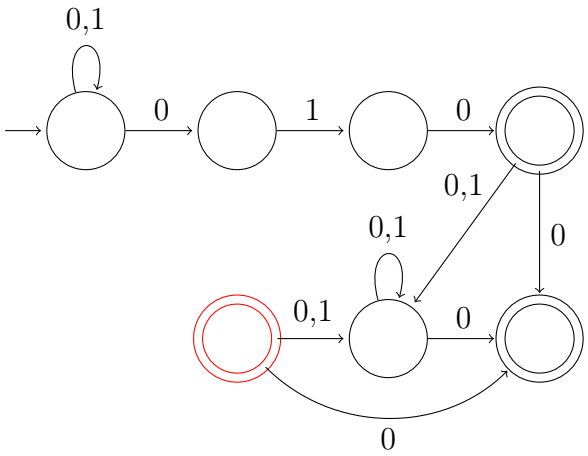
г) Автоматът г



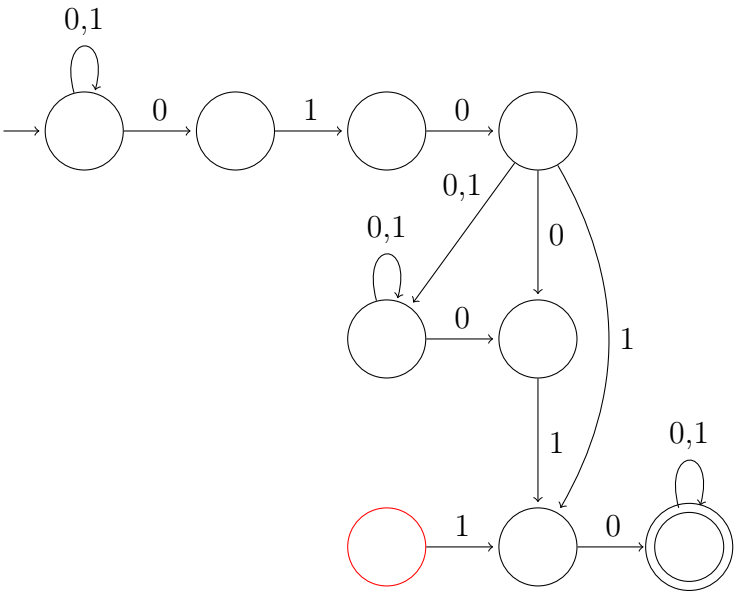
Сега следва да извършим операциите. Започваме с обединението на б и в:



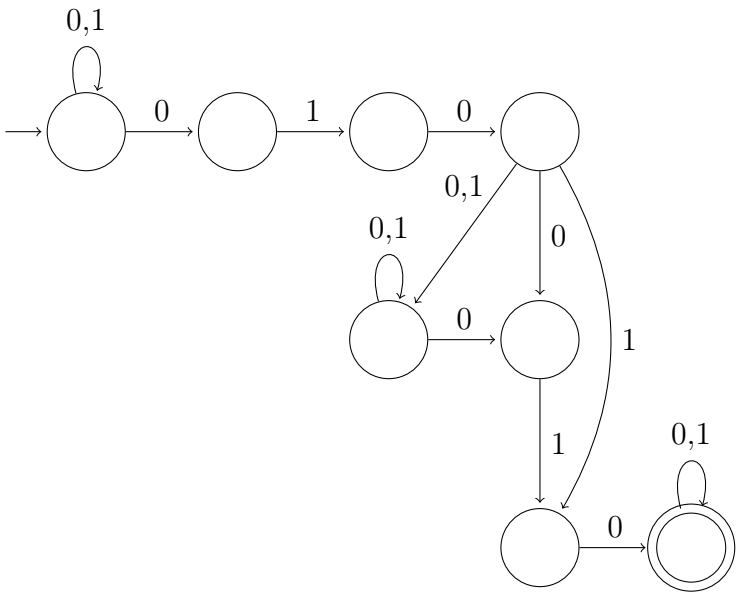
Виждаме, че тук е на лице недостижимо състояние, за това ще го махнем. Пристъпваме към конкатенация. Тя лесно може да се направи на една стъпка, но с цел да е по разбираемо ще го направим на две стъпки. Първо ще конкатенираме а с обединението на б и в:



Тук установяваме ново недостижимо състояние (маркирано с червено) и го премахваме. Сега правим конкатенация на новополучения автомат с автомата г:



Отново се появи недостижимо състояние (маркирано с червено). Като го премахнем получаваме крайния ни резултат:



Задачи Да разгледаме няколко задачи свързани с рекурентните зависимости:

Задача 1. (от контролно - 16.04.2015 г.) Да се състави регулярен израз за езика

$$\{uv : u \in \{a,b\}, v \in \{a,b\}^*, uv \text{ съдържа } uu \text{ като поддума}\}$$

На пръв поглед лесна задача. Имаме буква, след нея някаква поредица от други символи, а сред тях има последователност от два екземпляра на първата буква. Това лесно ще го опишем. Имаме две възможности за първа буква, които ще обединим:

$$a\{a,b\}^*aa\{a,b\}^* \cup b\{a,b\}^*bb\{a,b\}^*$$

А това дали са всички възможности? Нека видим следната примерна дума - *aababa*. Тя очевидно съдържа *aa* като поддума (префикс е в случая). А нашият регулярен израз хваща ли я? Отговорът е не. Защо? Ами погледнете първата част от регулярният ни израз. Тя изисква първият символ да е a и в останалата част да се намери последователността. А нашата дума няма такава последователност в часта след първия символ. Така освен разгледания преди малко случай се получава и един нов, в който думите започват с желаната последователност. Така изразът ни същност ще трябва да изглежда така:

$$a(a \cup \{a,b\}^*aa)\{a,b\}^* \cup b(b \cup \{a,b\}^*bb)\{a,b\}^*$$

Задача 2. Да се състави регулярен израз за език съдържащ всички битови низове с четен брой нули.

Битов низ означава низ от нули и единици. Този израз е лесен:

$$(1^*01^*01^*)^*$$

Това означава, че имаме някаква поредица от единици последвана от една нула, след това още единици и още една нула и после още единици. И това може да се повтаря много пъти. Всяко повторение ще съдържа по две нули и така ще имаме четен брой общо.

Задача 3. (от изпит - 25.08.2016 г.) Постройте тотален краен детерминиран автомат с език

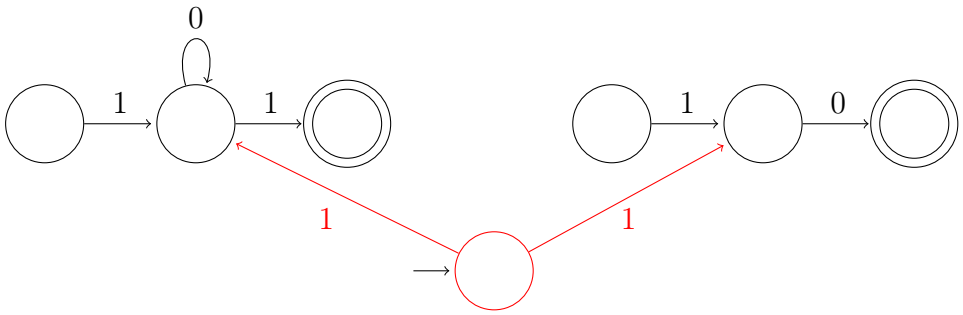
$$L = \{w \in \{a, b\}^* \mid \exists k \geq 0 \text{ r } w \text{ е двоичен запис на } 2^k + 1\}$$

След това минимизирайте построения автомат.

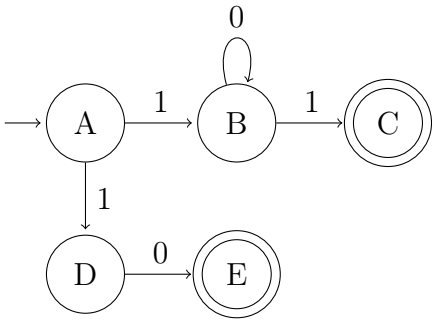
Ще решим задачата като първо съставим регулярен израз и после образуваме автомат по него. Нека започнем да гледаме как изглеждат двоичните записи за различните k:

- $k = 0 \Rightarrow 2^k + 1 = 2 = 10b$;
- $k = 1 \Rightarrow 2^k + 1 = 3 = 11b$;
- $k = 2 \Rightarrow 2^k + 1 = 5 = 101b$;
- $k = 3 \Rightarrow 2^k + 1 = 9 = 1001b$;
- $k = 4 \Rightarrow 2^k + 1 = 17 = 10001b$;
- ...

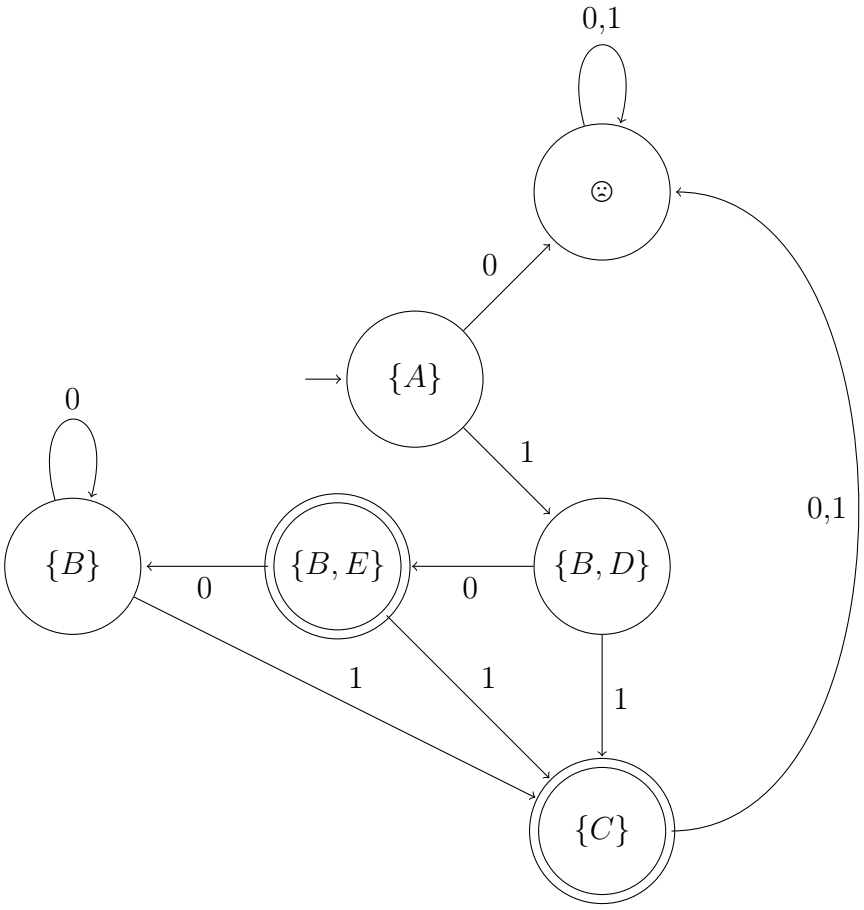
Тоест имаме $10^*1 \cup 10 \Leftrightarrow 1(0^*1 \cup 0)$. Нека сега съставим автомат по този регулярен израз. Ще имаме два прости автомата които после ще обединим (обединението е с червено):



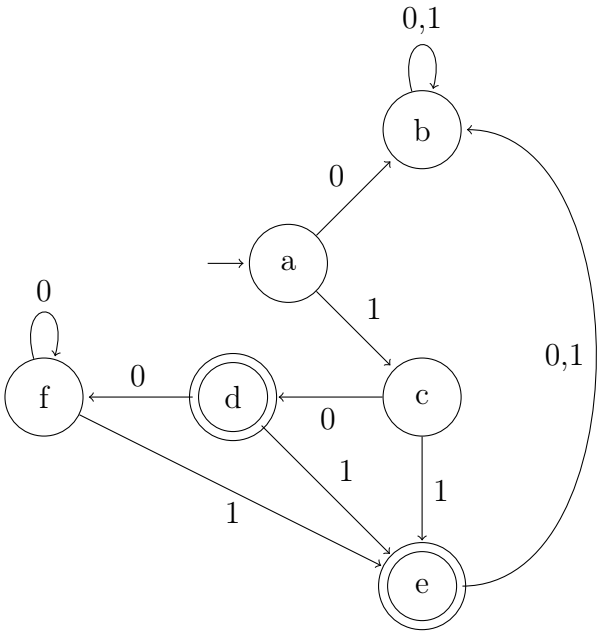
Сега като премахнем недостижимите състояния и дадем имена на състоянията ще получим следния автомат:



Сега следва да го детерминираме като същевременно ще го направим тотален. Правим това с директно приложение на алгоритъма, така че ще си спестя обясненията:



Сега остава само да минимизираме, но нека преди това преименуваме състоянията ни:



Сега да започнем минимизацията.

На първия етап имаме следното:

$$P_1 = \{a, b, e\}$$

$$P_2 = \{c, d\}$$

Сега навлизаме в итерационната част на алгоритъма:

Състояние	Преход с 0	Преход с 1
a	P_1	P_1
b	P_1	P_1
c	P_2	P_2
d	P_1	P_2
e	P_1	P_1
f	P_1	P_2

Веждаме, че и в двете множества имаме различни модели на поведени. Така се получават следните множества:

$$P_3 = \{a, e\}$$

$$P_4 = \{b\}$$

$$P_5 = \{c\}$$

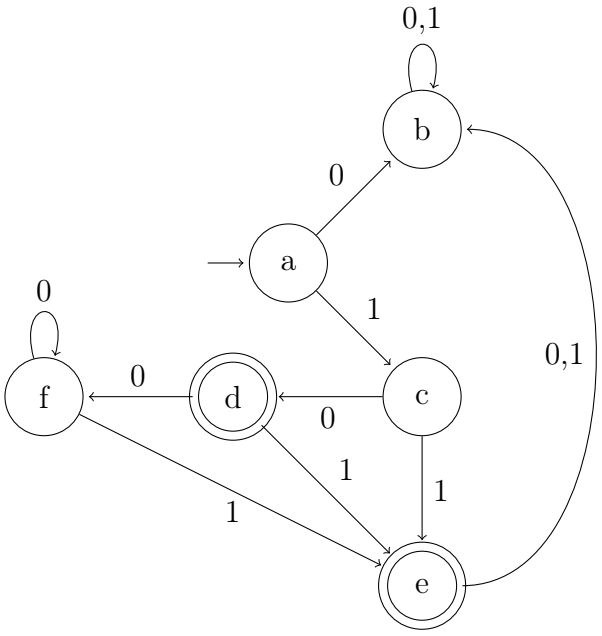
$$P_6 = \{d\}$$

$$P_7 = \{d\}$$

Отново изчисляваме таблицата:

Състояние	Преход с 0	Преход с 1
a	P_3	P_4
b	P_3	P_3
c	P_6	P_7
d	P_5	P_7
e	P_3	P_3
f	P_5	P_7

Виждаме, че и последното множество от повече от един елемент ще се разпадне. Това означава, че входният ни автомат е минимален. Тоест като краен отговор може да посочим:

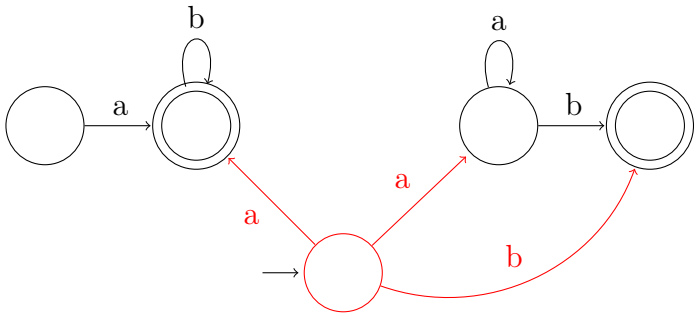


Задача 4. (от контролно - 11.04.2016 г.) Намерете недетерминиран краен автомат с език равен на този на регулярния израз

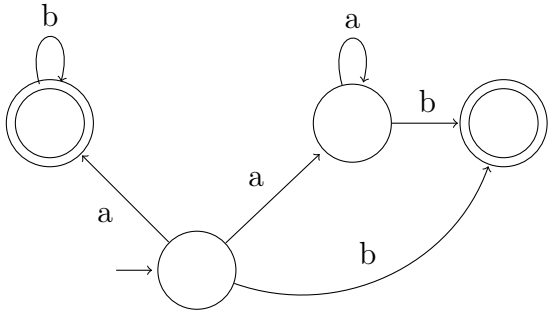
$$(ab^* \cup a^*b)^*$$

Започваме с конструирането на базовите ни автомати (ab^* и a^*b):

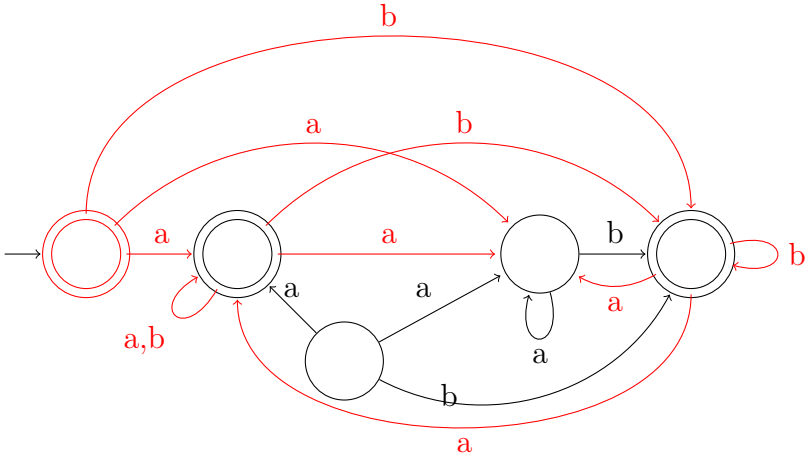
Сега да пристъпим към обединението (новото е с червено):



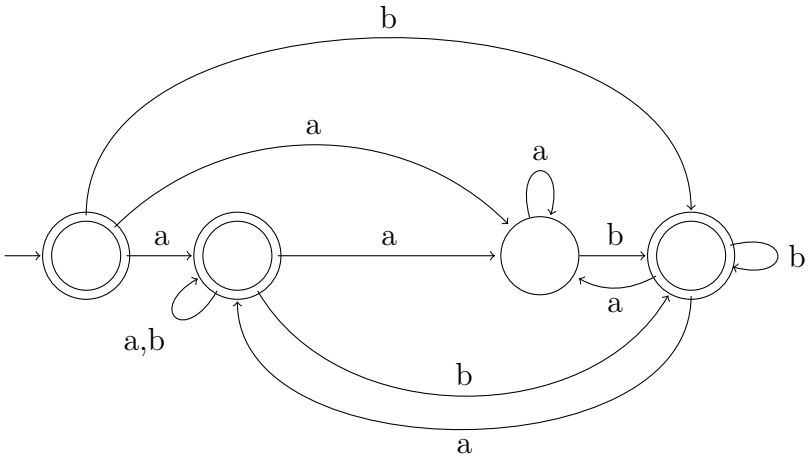
Тук получаваме недостижимо състояние, което премахваме и крайния резултат е:



Остава само да приложим звездата на Клини (новото е в червено):



Тук излиза ново недостижимо състояние, което ще премахнем. Като краен резултат получаваме:



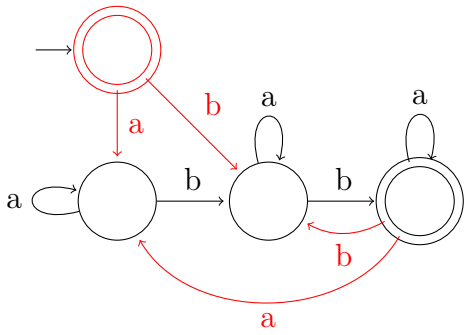
Задача 5. (от изпит - 04.07.2016 г.) Постройте минимален краен детерминиран автомат с език равен на:

$$L = \{w \in \{a, b, c\}^* \mid w \text{ започва или завършва с } a \text{ и } N_b(w) \equiv 0 \mod 2\}$$

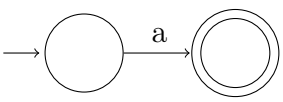
Нека първо съставим регулярен израз за езика:

$$a(a^*ba^*ba^*)^* \cup (a^*ba^*ba^*)^*a$$

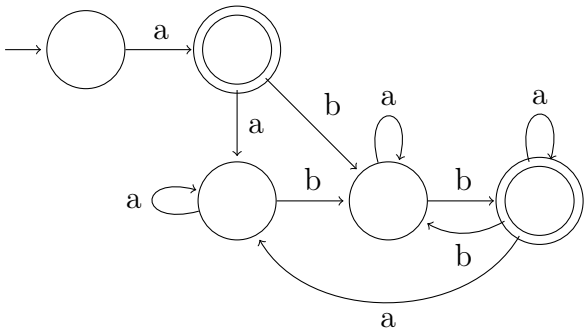
Това не е най - добрият израз, но е доста лесен за съставяне и мисля е подходящ в ситуация на контролно. Сега трябва да съставим автоматът за $(a^*ba^*ba^*)^*$. Ще съставим вътрешния автомат (с черен цвят) и после прилагаме звезда на Клини (с червено):

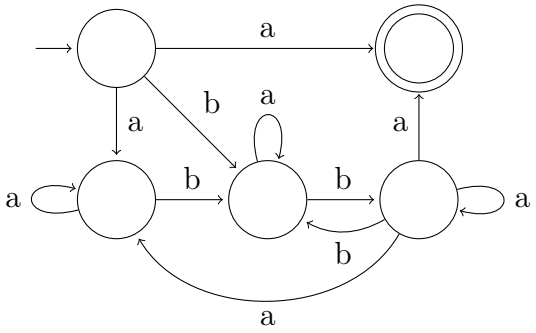


Сега правим и автомат за буквата a:

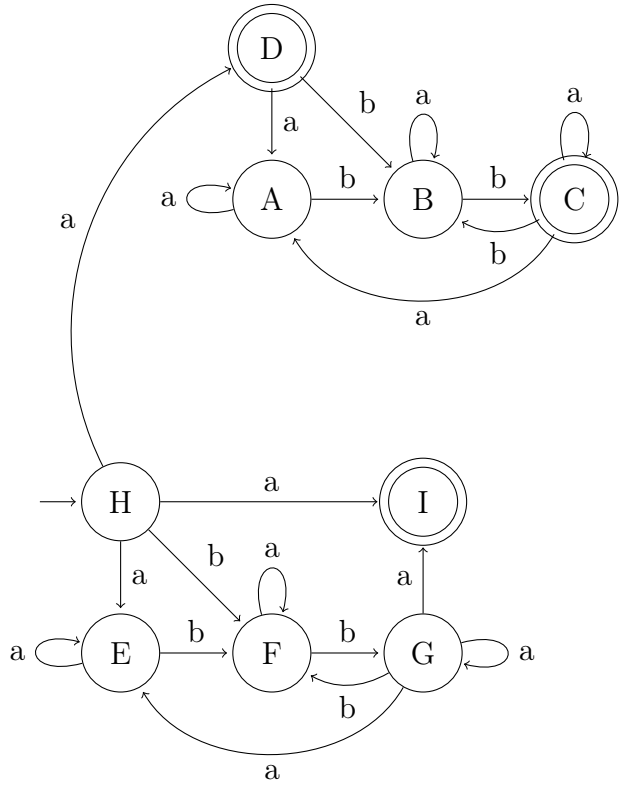


Сега имаме два автомата и следва да извършим конкатенация и в двете посоки. Първо правим $a(a^*ba^*ba^*)^*$ след това $(a^*ba^*ba^*)^*a$:

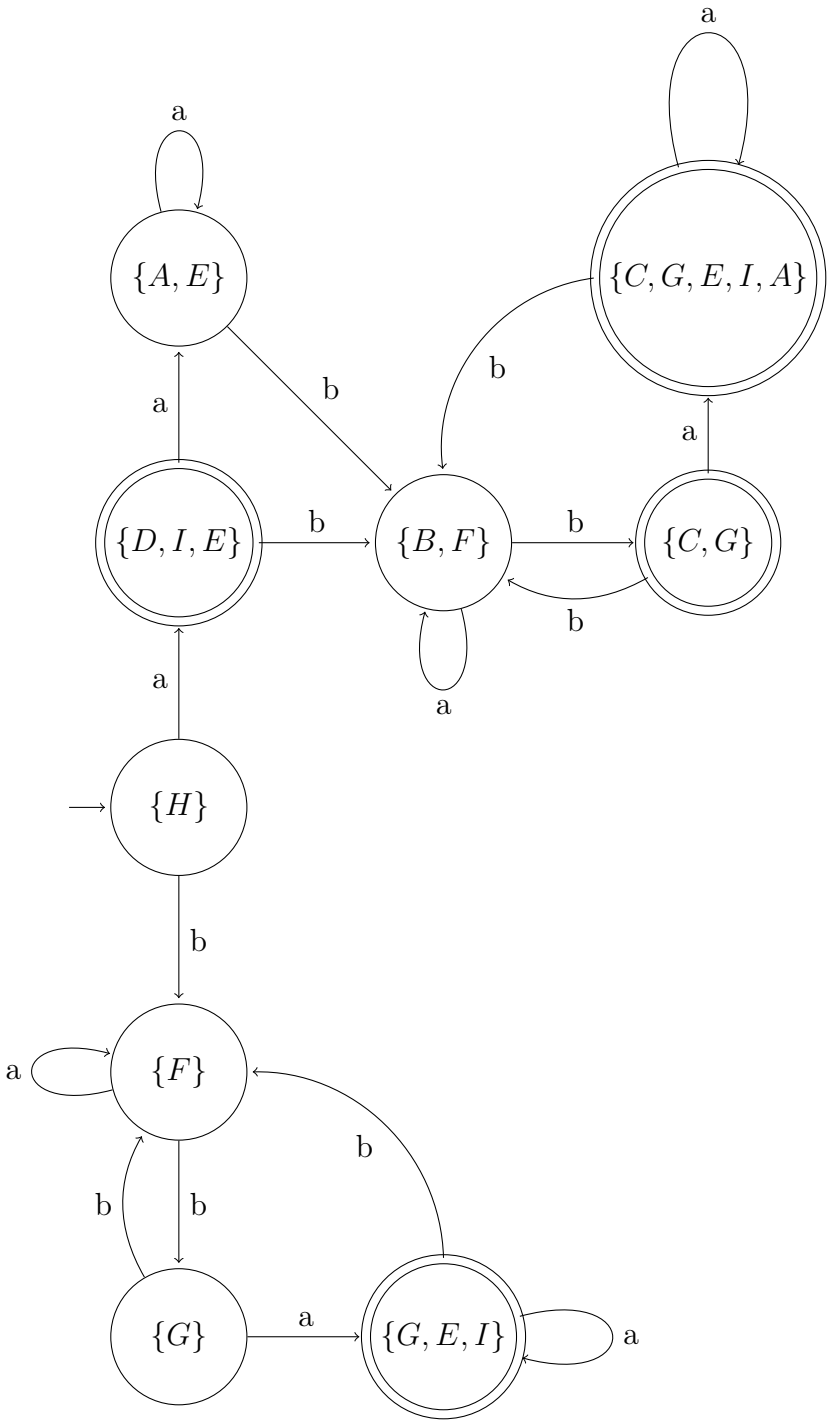




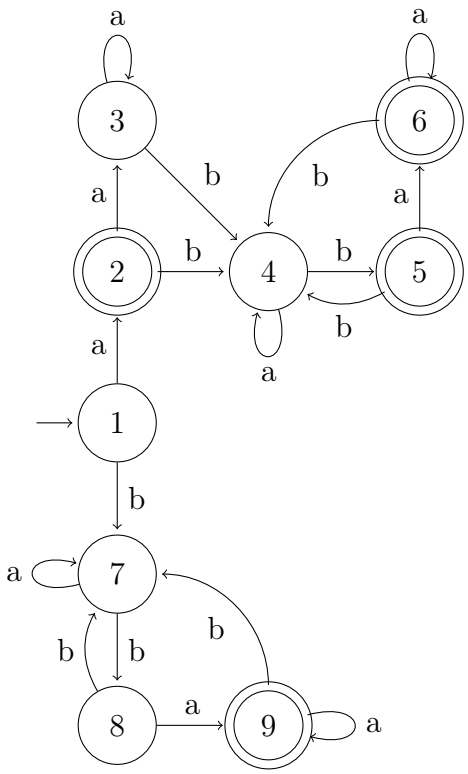
И сега ги обединяваме. Вече видяхме достатъчно примери за обединението, така че ще покажа директно резултата. Премахнал съм и недостижимите състояния:



Сега преминаваме към детерминизация:



Сега нека преименуваме състоянията:



И за финал остава да минимизираме автомата:

На първия етап имаме следното:

$$P_1 = \{1, 3, 4, 7, 8\}$$

$$P_2 = \{2, 5, 6, 9\}$$

Сега навлизаме в итерационната част на алгоритъма:

Състояние	Преход с a	Преход с b
1	P_2	P_1
2	P_1	P_1
3	P_1	P_1
4	P_1	P_2
5	P_2	P_1
6	P_2	P_1
7	P_1	P_1
8	P_2	P_1
9	P_2	P_1

В първото множество се наблюдават три различни модела на поведение, а във втория - два. Така се получават следните множества:

$$P_3 = \{1, 8\}$$

$$P_4 = \{3, 7\}$$
$$P_5 = \{4\}$$
$$P_6 = \{2\}$$
$$P_7 = \{5, 6, 9\}$$

Отново изчисляваме таблицата:

Състояние	Преход с a	Преход с b
1	P_6	P_4
2	P_4	P_5
3	P_4	P_5
4	P_5	P_7
5	P_7	P_5
6	P_7	P_5
7	P_4	P_3
8	P_7	P_4
9	P_7	P_4

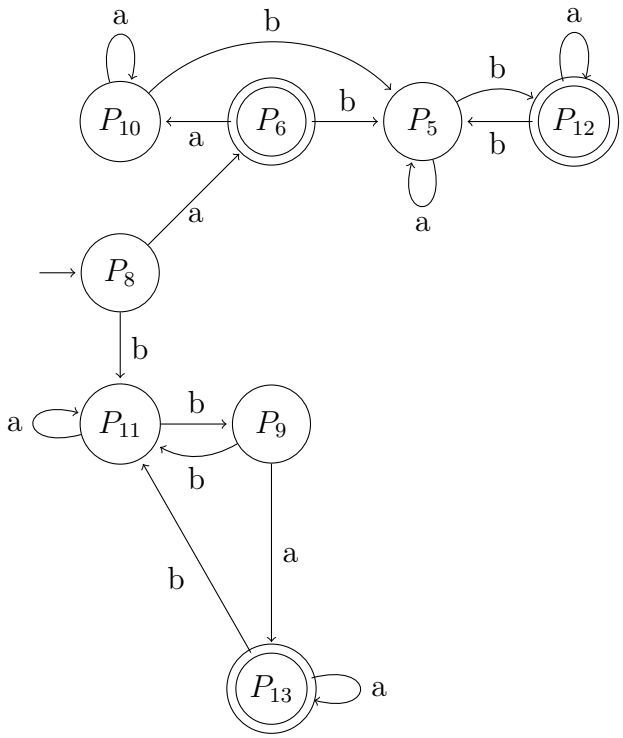
Тук настъпват нови разпади и получаваме:

$$P_5 = \{4\}$$
$$P_6 = \{2\}$$
$$P_8 = \{1\}$$
$$P_9 = \{8\}$$
$$P_{10} = \{3\}$$
$$P_{11} = \{7\}$$
$$P_{12} = \{5, 6\}$$
$$P_{13} = \{9\}$$

Отново изчисляваме таблицата:

Състояние	Преход с a	Преход с b
1	P_6	P_{11}
2	P_{10}	P_5
3	P_{10}	P_5
4	P_5	P_{12}
5	P_{12}	P_5
6	P_{12}	P_5
7	P_{11}	P_9
8	P_{13}	P_{11}
9	P_{13}	P_{11}

От тази таблица се вижда, че няма да има повече разбивания, така получаваме следния резултатен автомат:



Глава 6

Нерегулярни езици

Общи сведения Задачите свързани с доказване, че един език не е регулярен се свеждат до Лемата за разрастването. Тя гласи следното: За всеки регулярен език L, съществува естествено число N, такова че за всяка дума w от L с дължина поне N, съществуват думи x, y, z, за които $w = xyz$, $|xy| \leq N$ и $y \neq \varepsilon$ и за всяко естествено число i, думата xy^iz е от L.

За да решим задача, в която се изисква да докажем, че даден език не е регулярен е достатъчно да допуснем, че е регулярен, да проследим теоремата като в зелените части избираме конкретни примери, чрез които накрая да достигнем до противоречие.

Задачи Да разгледаме няколко задачи свързани с Лемата:

Задача 1. (от контролно - 24.04.2014 г.) Докажете, че езикът L не е регулярен, където L е:

$$L = \{w \in \{1\}^* \mid w = 1^{2p}, p \text{ е просто число}\}$$

Допускаме, че L е регулярен. Да приложим лемата за L. От лемата получаваме някакво естествено число N. Нека $w = 1^{2k}, k > N, k$ е просто . От лемата получаваме три думи x, y, z, където $w = 1^{2k} = xyz, y \neq \varepsilon, |xy| \leq N$. От това директно следва, че думите x, y, z са съставени само от 1. Тогава за някои i, j, h е в сила: $x = 1^i, y = 1^j, z = 1^h$. От $w = 1^{2k}$ следва, че $i + j + h = 2k$. Освен това $j \neq 0$ и $k + j < N$. От лемата следва, че за всяко естествено число c, $\dfrac{i + c * j + h}{2}$ е просто число. Нека $c = 2k + 4j + 5$. Получаваме:

$$\frac{i + (2k + 4j + 5) * j + h}{2} = \frac{(2k + 4j + 5) * j - j + 2k}{2} =$$

$$= \frac{(2k+4j+5-1)*j+2k}{2} = \frac{(2k+4j+4)*j+2k}{2} =$$
$$= (k+2j+2)*j+k = (k+2j)*j+(2*j+k) = (k+2*j)(j+1)$$

От факта, че $j \neq 0$ следва, че това е произведение от две числа по - големи от 1. Това не може да е просто число. Така стигаме до противоречие, следователно езика не е регулярен.

Задача 2. (от изпит - 04.07.2016 г.) Докажете, че езикът L не е регулярен, където L е:

$$L = \{w \in \{a,b\}^* \mid 3N_a(w) < 2N_b(w) + 1\}$$

Допускаме, че L е регулярен. Да приложим лемата за L. От лемата получаваме някакво естествено число N. Нека $w = a^{2N}b^{3N}$. От лемата получаваме три думи x, y, z, където $w = a^{2N}b^{3N} = xyz, y \neq \varepsilon, |xy| \leq N$. От това директно следва, че думите x, y са съставени само от a. Тогава за някои k, l е в сила: $x = a^k, y = a^l$. От това следва, че $z = a^{2N-k-l}b^{3N}$. От лемата следва, че за всяко естествено число i, $w' = xy^iz \in L$, тоест $3N_a(w') < 2N_b(w') + 1$. Нека $i = 2$. Получаваме:

$$3N_a(w') = 3(k + i*l + 2N - k - l) = 6N + 3(2-1)l = 6N + 3l$$

$$2N_b(w') + 1 = 6N + 1$$

Тъй като $l > 0$, се получава $3N_a(w') > 2N_b(w') + 1$. Така стигаме до противоречие, следователно езика не е регулярен.

Задача 3. (от изпит - 02.07.2014 г.) Докажете, че езикът L не е регулярен, където L е:

$$L = \{vuv \mid u, v \in \{a,b\}^*, |u| < |v|\}$$

Допускаме, че L е регулярен. Да приложим лемата за L. От лемата получаваме някакво естествено число N. Нека $w = a^{2N}ba^{2N}$. От лемата получаваме три думи x, y, z, където $w = a^{2N}ba^{2N} = xyz, y \neq \varepsilon, |xy| \leq N$. От това директно следва, че думите x, y са съставени само от a. Тогава за някои k, l е в сила: $x = a^k, y = a^l$. От това следва, че $z = a^{2N-k-l}ba^{2N}$. От лемата следва, че за всяко естествено число i, $w' = xy^iz \in L$, тоест от двете страни на средната дума да стоят еднакви думи и въпросната средна дума да е по - къса от страничните. Нека $i = 2N+1$. Получаваме:

$$w' = a^{k+i*l+2N-k-l}ba^{2N} = a^{2N+2N*l}ba^{2N}$$

Тъй като имаме само едно b то задължително трябва да е в средната дума. Нека първо разгледаме минималната възможна средна дума: $u = a^{2N*l}b$, тъй като $l > 0$, u е размер по - голям от $v = a^{2N}$. Което води до противоречие. Възможно е да взимаме по едно a от страничните думи и да ги добавяме към средната, но това само ще влоши ситуацията. Така получихме, че езикът не е регулярен.

Задача 4. (от изпит - 02.07.2014 г.) Докажете, че езикът L не е регулярен, където L е:

$$L = \{w \in \{a, b\}^* \mid w \neq w^R\}$$

Тъй като регулярността се запазва при отрицанието, всъщност можем да докажем, че следният език не е регулярен:

$$L_1 = \{w \in \{a, b\}^* \mid w = w^R\}$$

Тъй като $\overline{L_1} = L$, остава само да докажем, че L_1 не е регулярен. Допускаме, че L_1 е регулярен. Да приложим лемата за L_1. От лемата получаваме някакво естествено число N. Нека $w = a^N ba^N$. От лемата получаваме три думи x, y, z, където $w = a^N ba^N = xyz, y \neq \varepsilon, |xy| \leq N$. От това директно следва, че думите x, y са съставени само от a. Тогава за някои k, l е в сила: $x = a^k, y = a^l$. От това следва, че $z = a^{N-k-l}ba^N$. От лемата следва, че за всяко естествено число i, $w' = xy^iz \in L_1$, тоест $k+i*l+N-k-l = N$. Нека $i = 2$. Получаваме:

$$N + l = N, l > 0$$

Това е противоречие. Следователно L_1 не е регулярен, от това следва, че $L = \overline{L_1}$ не е регулярен.

Задача 5. (класически пример - Pumping down) Докажете, че езикът L не е регулярен, където L е:

$$L = \{0^h 1^j \mid h > j\}$$

Допускаме, че L е регулярен. Да приложим лемата за L. От лемата получаваме някакво естествено число N. Нека $w = 0^{N+1}1^N$. От лемата получаваме три думи x, y, z, където $w = xyz, y \neq \varepsilon, |xy| \leq N$. От това директно следва, че думите x, y са съставени само от 0. Тогава за някои k, l е в сила: $x = 0^k, y = 0^l$. От това следва, че $z = 0^{N-k-l}1^N$. От лемата следва, че за всяко естествено число i, $w' = xy^iz \in L$, тоест $k + i * l + N - k - l > N$. Нека $i = 0$. Получаваме:

$$N - l > N, l > 0$$

Това е противоречие. Следователно L не е регулярен.