

Teoretyczne

Podstawy

Informatyki

TP I

$s: X \rightarrow Y$

$f(x) = ?$

$x \in X$

$\alpha(f(x)) \rightarrow X$

$y \in Y$

$\hat{\alpha}$

algorytm

zadanie

algorytm może wykonać zadanie  
na wiele różnych sposobów.

$$\|\alpha(I(x)) - s(x)\| = \varepsilon(x) \quad \text{Stan końcowy}$$

zbior symboli algorytm precjacji

$$MT = \{Q, P, \Sigma, h, G, \rho_0, F\}$$

skończony zbiór stanów

stan  
początkowy

MT ma wirtualna taśma

#	0	1	1	0	1	0	1	0	#
---	---	---	---	---	---	---	---	---	---

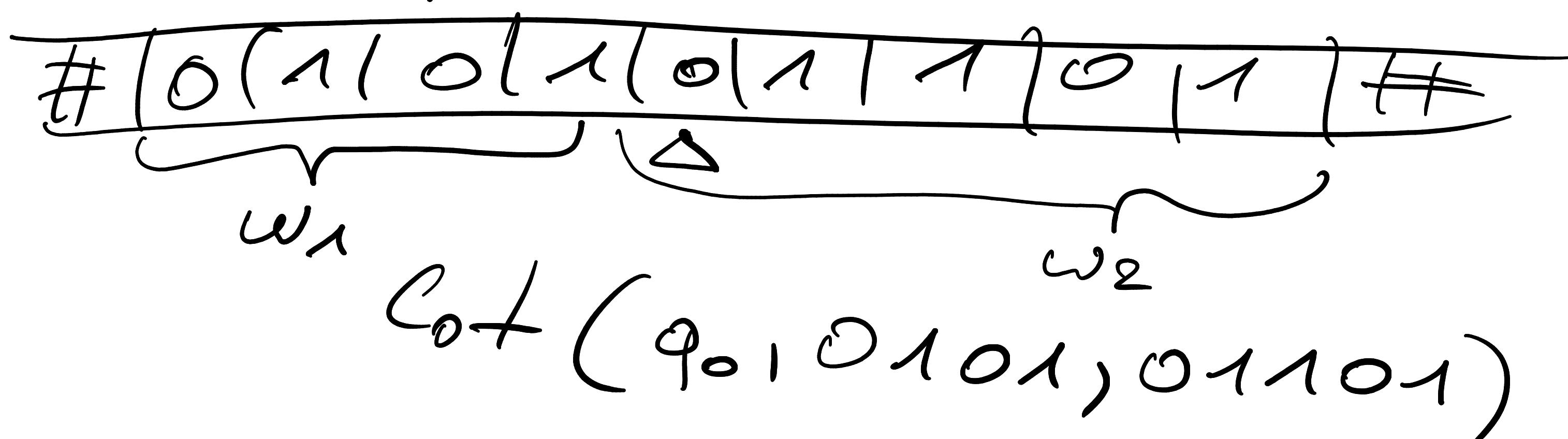
# - symbol pusty,  $\Gamma = \{\#\}$ ,  $\Sigma = \{0, 1\}$

reszta jest ciągłe pusta

Głowica domyslnie zaczyna na  
pierwszym niepustym miejscu w pamięci.

Konfiguracja maszyny Turinga (3F):

$$1. C_i = (q_i, w_1, w_2)$$



Zadanie 1.

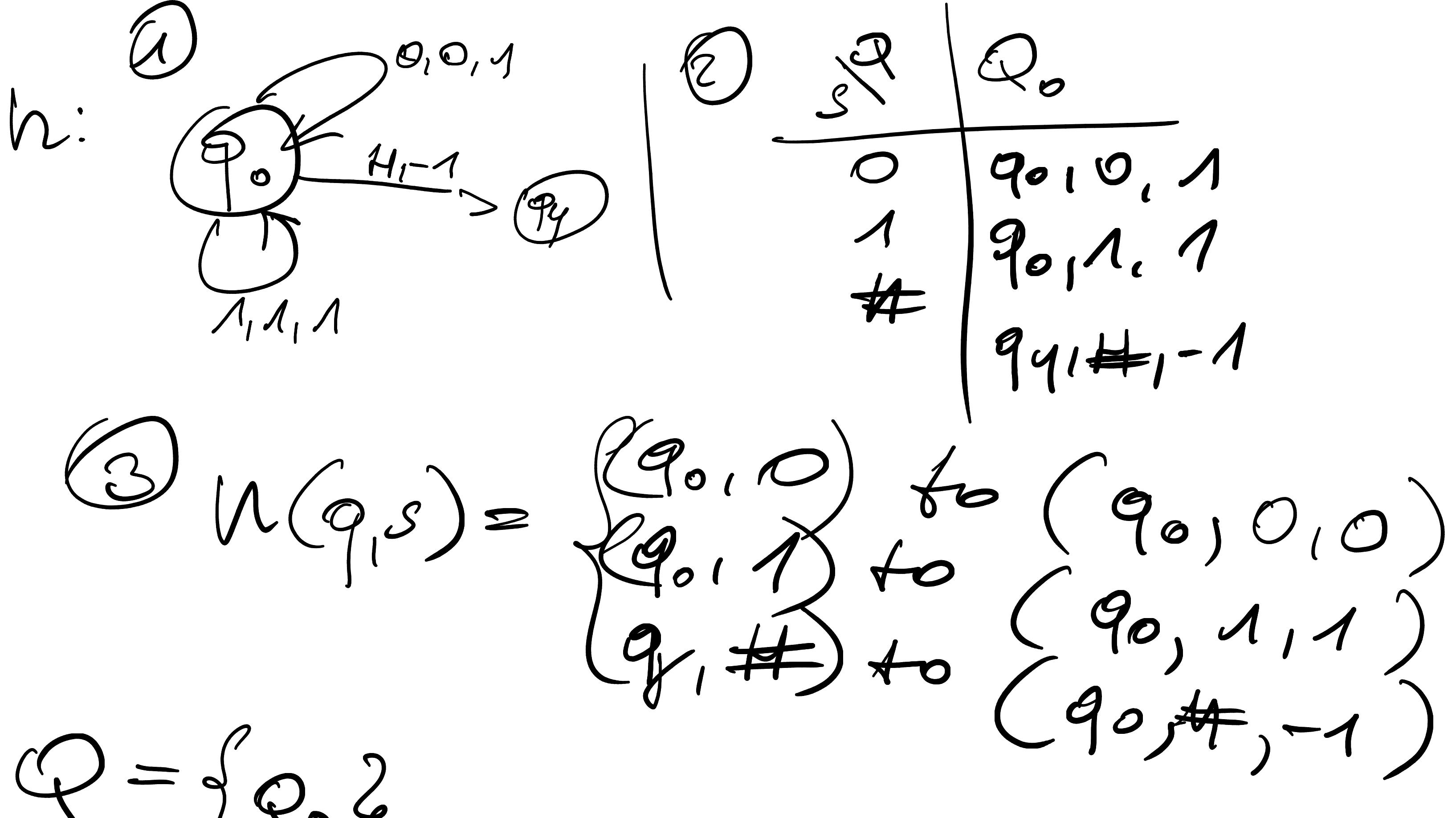
Stwórz deterministyczną maszynę Turinga (DNT),  
która dla alfabetu  $\Sigma = \{0, 1\}$ ,  
przesiedzie głosice na ostatni  
symbol słowa.

np. #  $\begin{matrix} 0 \\ \Delta \end{matrix}$  1 1 0 1 1 #

$$\lambda: Q \times T \rightarrow Q \times T' \times \{-1, 0, 1\}$$
$$\hookrightarrow \{q, \delta\} = (q', \delta')$$

fasina nie może być pusta, bo algorytm może działać tylko na określonych danych

Maszyna zatrzymuje się na symbolach, których nie ma.



$$Q = \{Q_0\}$$

$$\Gamma = \{0, 1, \#\}$$

$$\Sigma = \{0, 1\}$$

$$h: \uparrow$$

$$G = \{\neq\}$$

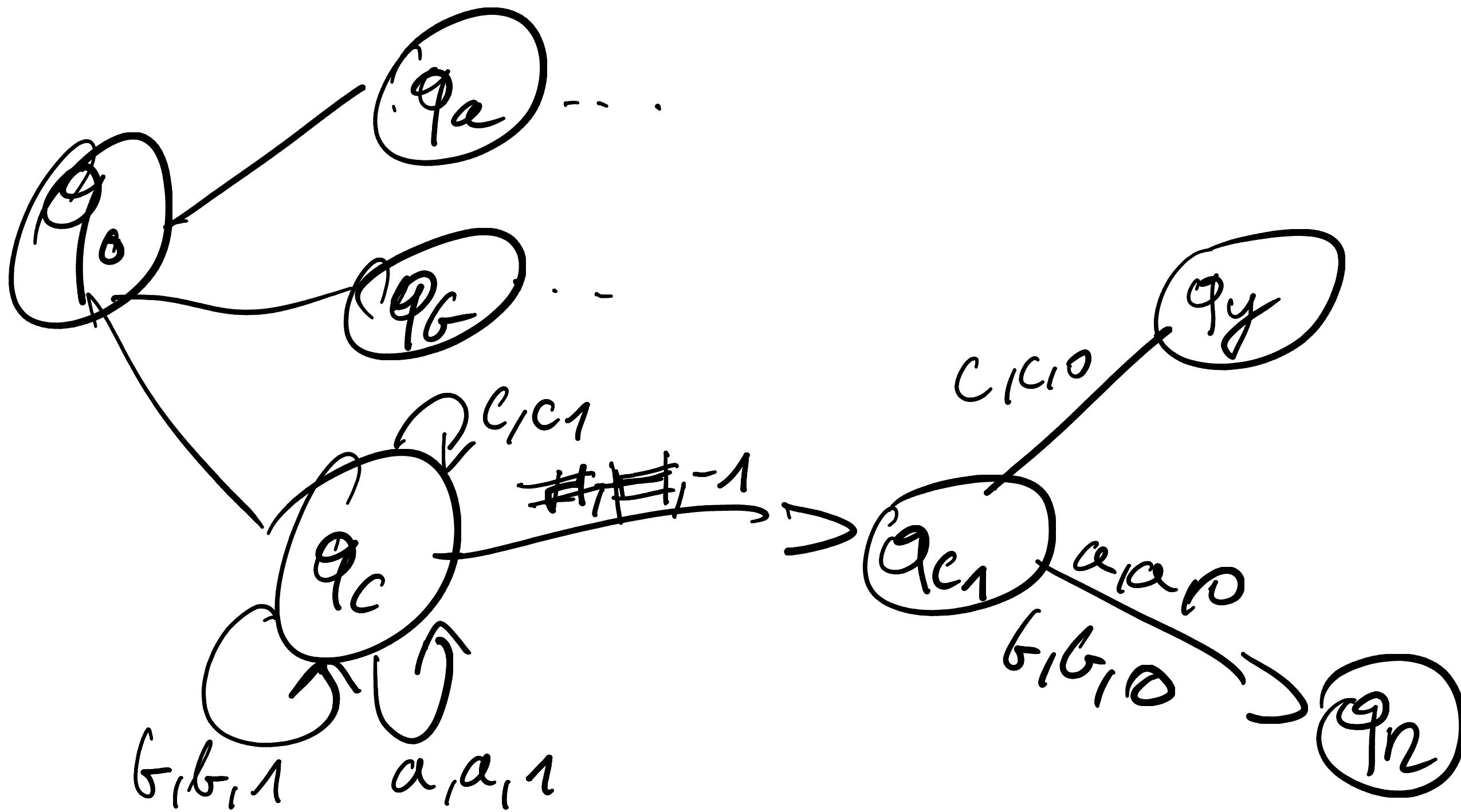
$$q_0 = \{q_0\}$$

$$F = \{q_1\}$$

Zad. 2. Ułóż oparte na alfabetie

$\Sigma = \{A, B, C\}$ . Znaleźć DMT czy

pierwszy i ostatni symbol na  
taśmie jest identyczny.



Zad. 3. Dla stocia opartego na alfabetie  $\alpha, \beta, \gamma$  zbuduj DLT  
która unie wystąpienie symbolu  $\beta$ .

$\# A, B, C, A, B, C \#$

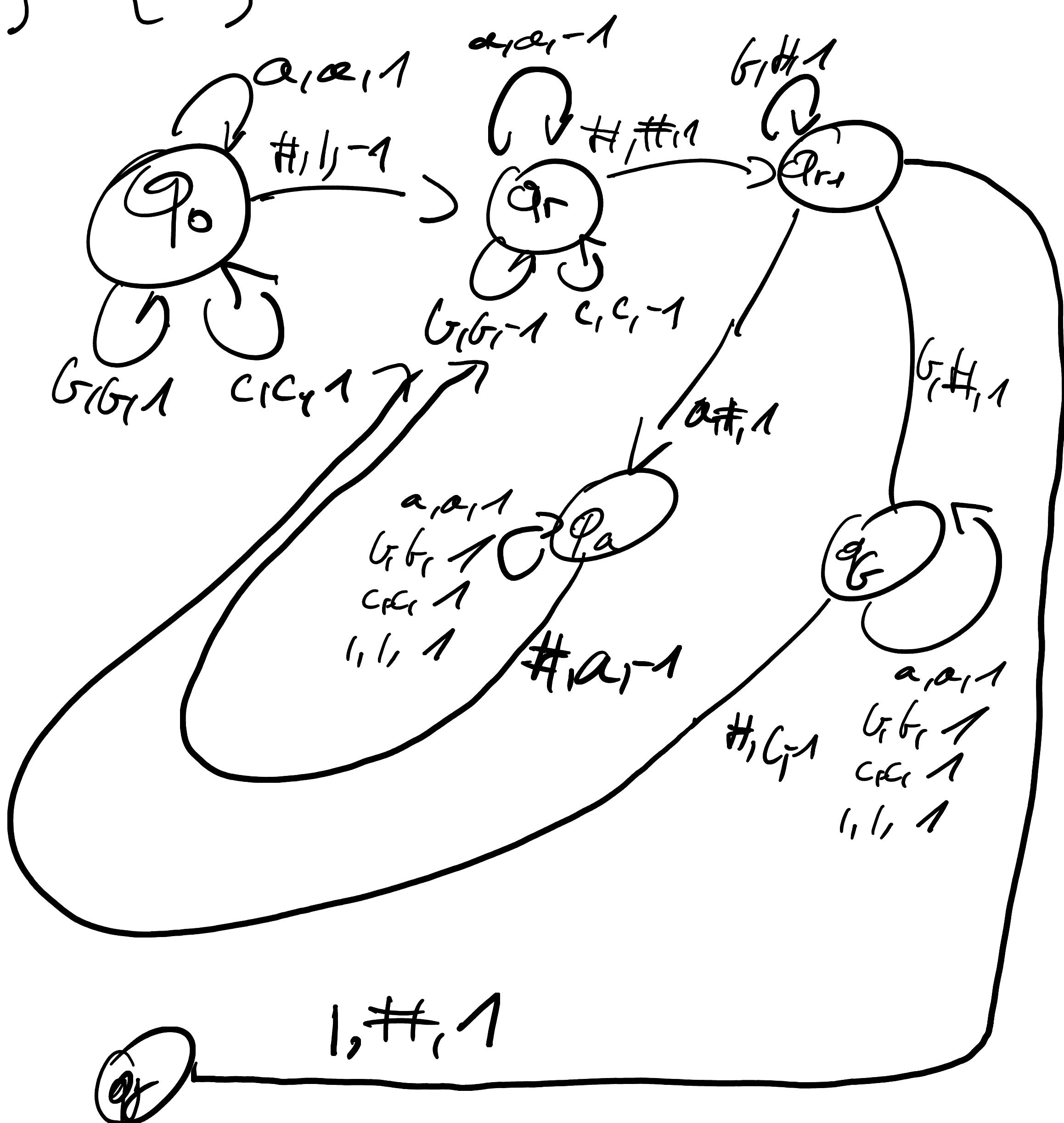
$Z = \{A, B, C\}$

$\Gamma = Z \cup \{\#\} \cup \{\}\}$

~~#abc #~~  
~~#abc/c #~~  
~~#bc/a #~~  
~~##bc/a #~~  
~~##/c/a #~~  
~~##/a/c #~~  
~~#ac#~~

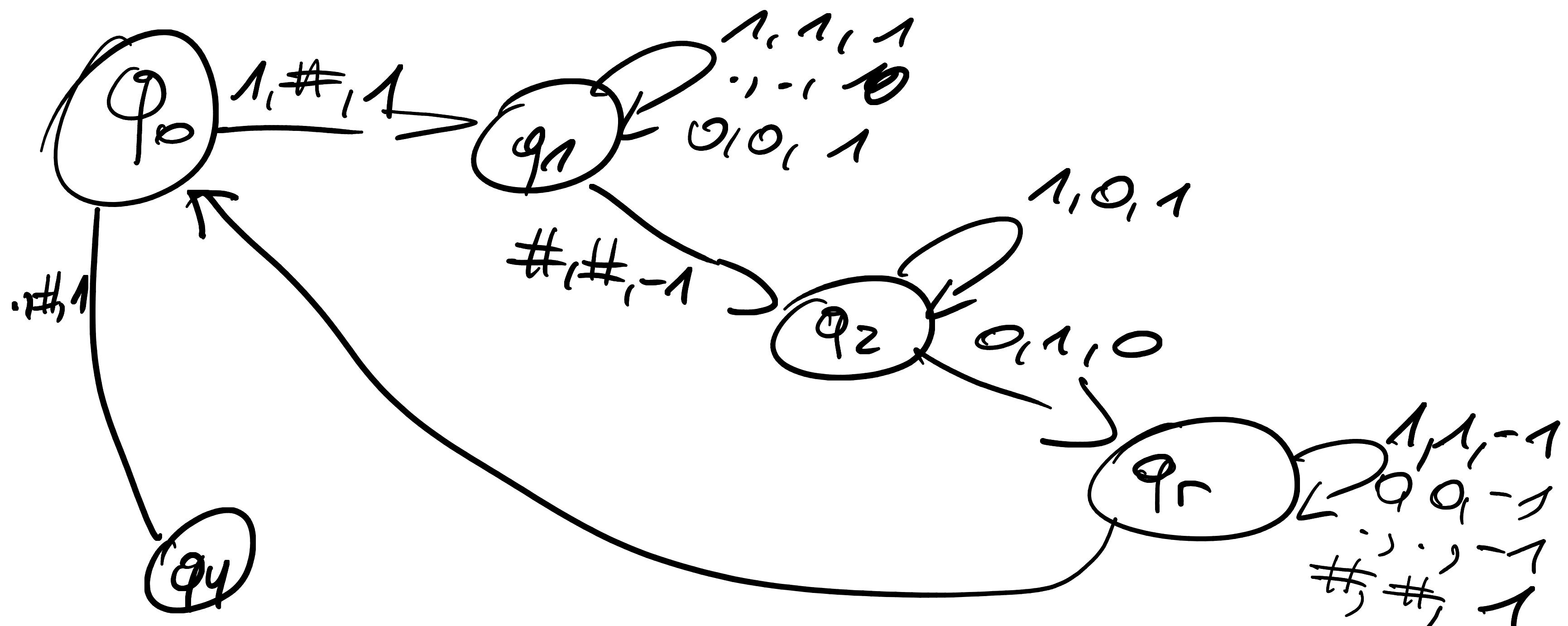
$S_1 1$  = litera alternatywa

$S_2 1$  = litera koniunkcja



Zad. 6. Znaleźć DNF utoższy policyjne dla jest zaszyfrowanej

# 1 111. 00 00 #



Zad. 5.

$$\sum \{a, b, c\}$$

Dla  $\alpha$  zapisz w kolejności alfabetu {a, b, c} wypisany串の順序でアルファベットの順序で書く。

$$\# a, b, c, a, b, \#$$

Takie się zapisy daje 2 do F.

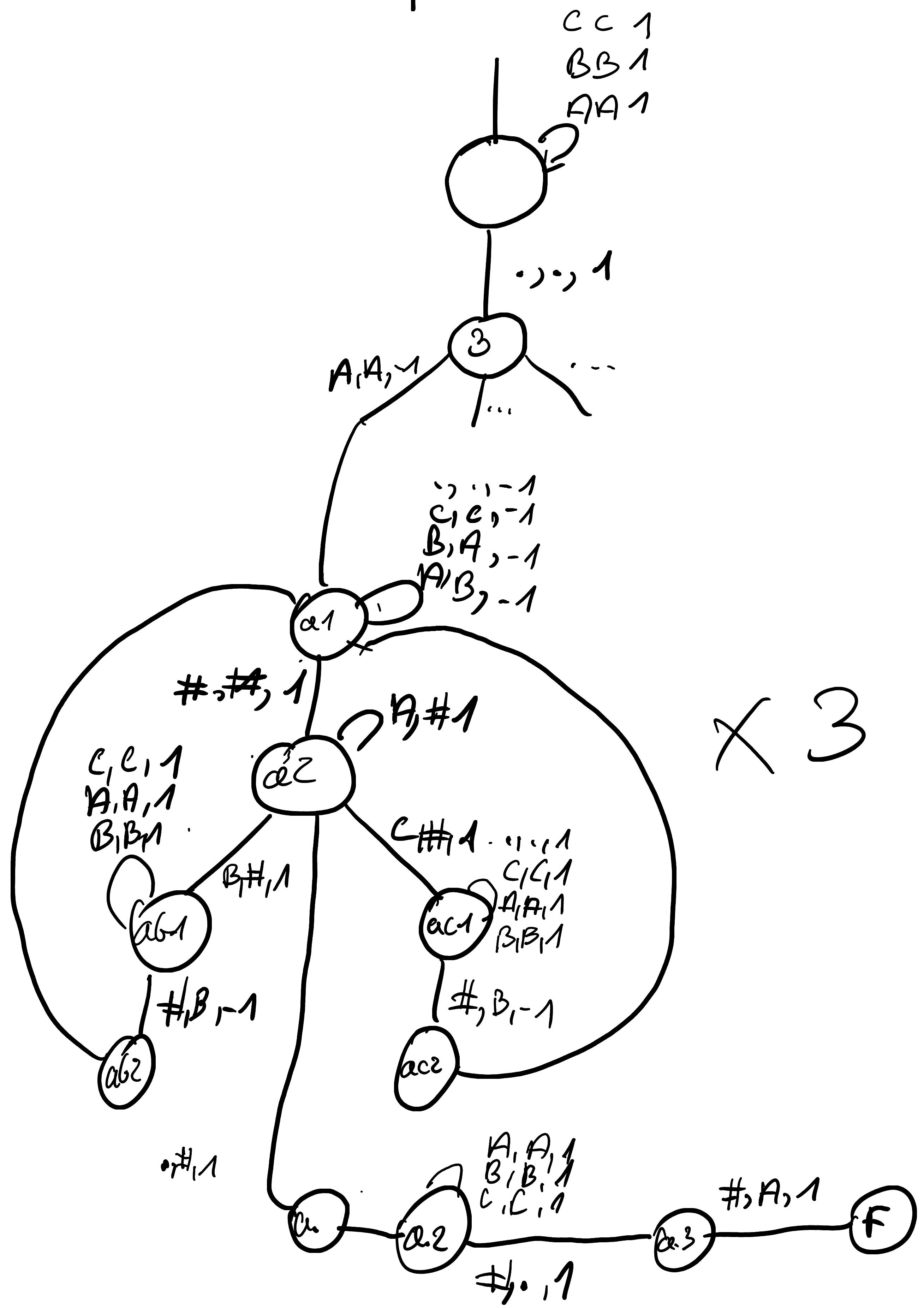
$$\sum \{a, b, c\}$$

#abcabc. ab #  
w<sub>1</sub> w<sub>2</sub>

Usunię wszystkie w2 z w1.

- 1) 1 znak, zostawiający #bbb. a#
- 3) Usunąć z znaku

$A \ B \ C, C_1$



Zadanie 1

$n^{\lg(n)}$ ,  $\lg(n!)$ , ... Wprowadzić w kolejności

Zadanie 2.

Na pewnym komputerze wykonanie algorytmu o złożoności  $T(n) = 2^n$  dla danych o rozmiarze  $n = 6$  zajmuje 8 sek.

a) Wyznaczyć dla  $n = 10$   $2^7 = 128$

b) natwierdzić rozmiar dla  $s < 32$   $n = 8$

c) po skopiowaniu 1024 razy,  $n = 20$  128s

Zadanie 3.

a)  $\sum_{i=1}^n i^k$  ma stopniowość  $O(n^{k+1})$ ,  $k > 0$

$$\sum_{i=1}^n i^k = 1^k + 2^k + \dots + (n-1)^k + n^k \Rightarrow$$
$$n^k + n^k + n^k + \dots + n^k = n n^k = n^{k+1}$$

b)  $\frac{an^k}{\log(n)}$  ma stopniowość  $O(n^k)$

c)  $2^n$  ma stopniowość  $O(n!)$

Zadanie 4

Oceńić prawdziwość zdań

Wyznaczanie złożoności czasowej.

```
for (i = sum = 0; i < n; i++) {  
    sum = sum + a[i]  
}
```

Liczba przypisów w pętli:

$$2 + \sum_{n=0}^{n-1} 2 = 2 + 2n$$

Złożoność:  $T(n) = O(n)$

```
for (i=0; i<n; i++) {
```

```
    for (j=1, sum=a[0], j<=i; j++)  
        sum = sum + a[j]
```

```
    print(...)
```

$$1 + \sum_{i=0}^{n-1} \left( 3 + \sum_{j=1}^i 2 \right) = 1 + 3n + 2(1+2+\dots+n-1) =$$

$$= 1 + 3n + n(n-1) = \mathcal{O}(n^2)$$

# Metoda iteracyjna

MD, patrząc, powtarza się  
Schemat, zapisujemy  $\Sigma$

## Metoda rekurencyjna, uogólniona

1. Jeśli  $f(n) = O(n^{\log_b a - \varepsilon})$

stąd  $\varepsilon$        $T(n) = \Theta(n^{\log_b a})$       dla pewnej

2. Jeśli  $f(n) = \Theta(n^{\log_b a})$  to

$$T(n) = \Theta(n^{\log_b a} \log n)$$

3. Jeśli  $f(n) = \Omega(n^{\log_b a + \varepsilon})$  to

$$T(n) = \Theta(f(n))$$

$$\text{np. : } T(n) = T(2n/3) + 1$$

$$\alpha = 1, \beta = \frac{3}{2}, n^{\log_3 1} = n^{\log_{3/2} 1} = n^0 = 1$$

Przykład 2.

$$T(n) = 3T(n/4) + n \log(n)$$

Przykład 3.  $\alpha=2, \beta=4$

$$f(n) = n \log(n)$$

$$T(a) = 2T(\frac{a}{2}) + n \log(a)$$

Występuje między 2 a 3. Taka iteracyjnie. Stosunek  $f(n)/n^{\log_2 2} =$   
 jest wielokrotnością  $n^\epsilon$  dla każdego  $\epsilon < 0$

Klasy stwarzające problemy edycji danych.

Klasa P - można rozwiązać w  
akceptowalnym czasie używając  
współczesnych PC.