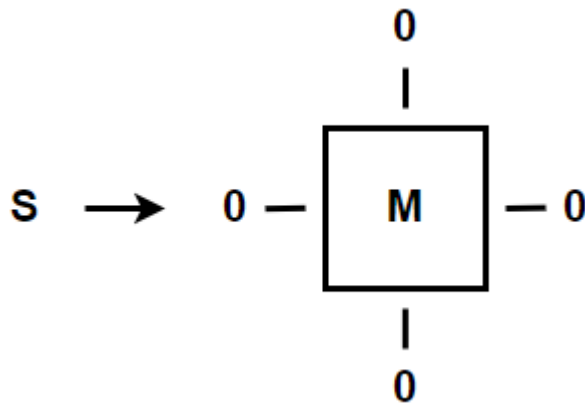


Sprawozdanie cw4

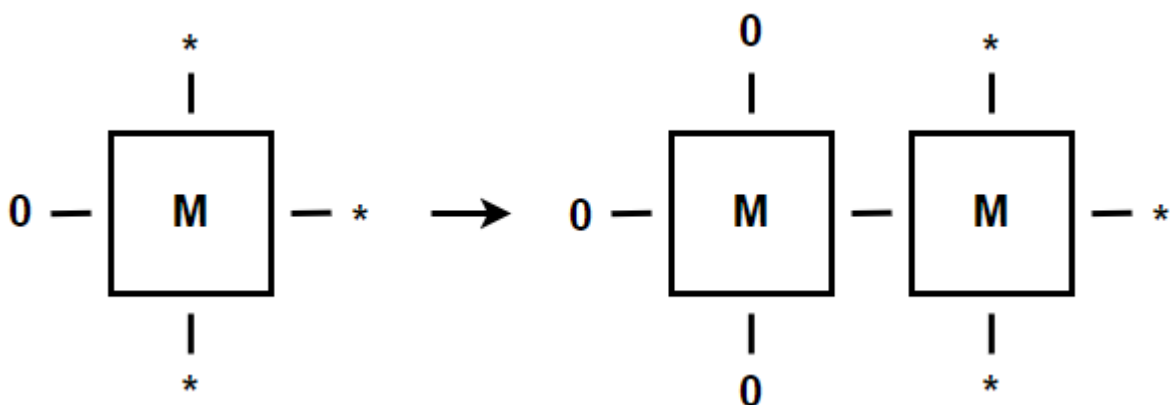
Mateusz Nowak

Produkcje gramatyki

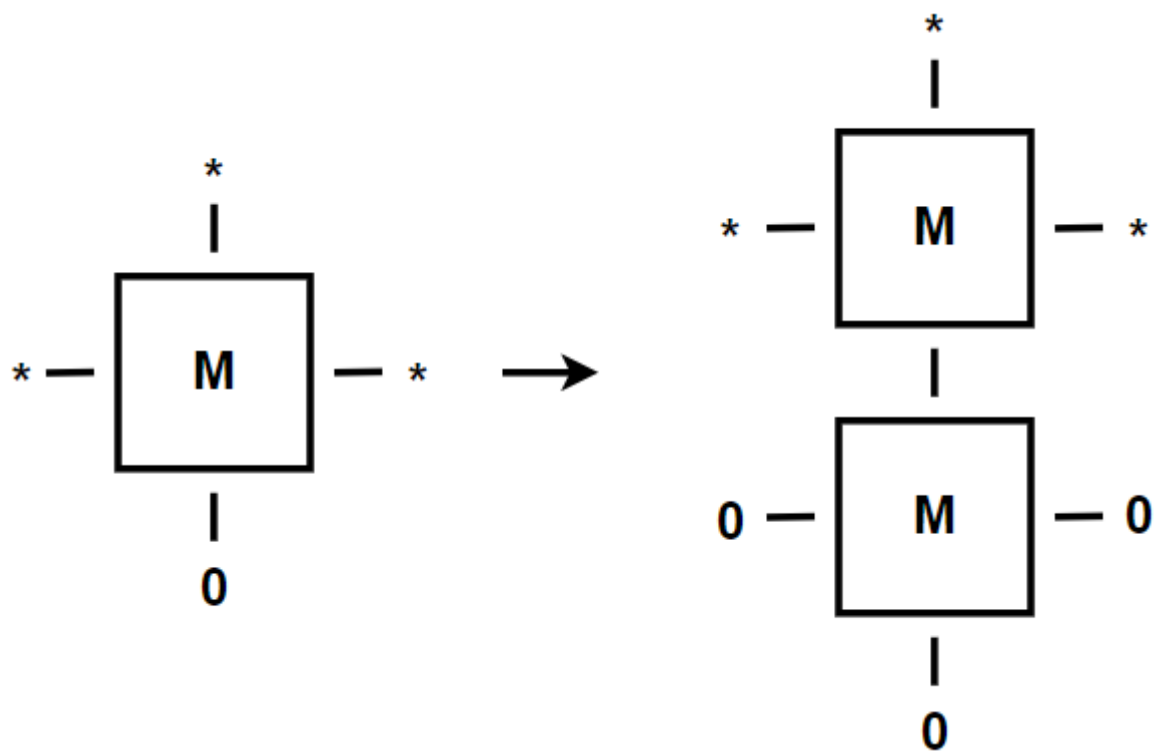
PI - produkcja wyjściowa z S generuje pojedynczy element siatki



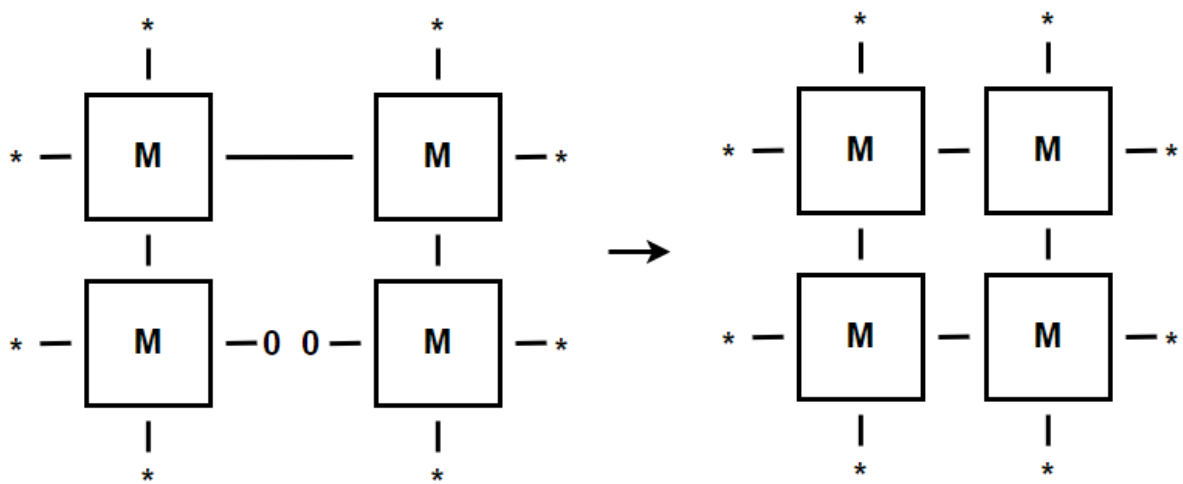
PW - produkcja generująca sąsiada z lewej strony (West)



PS - produkcja generująca sąsiada od dołu (South)



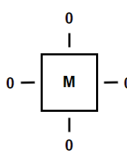
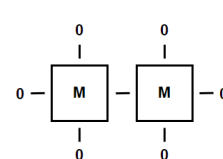
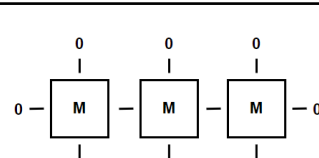
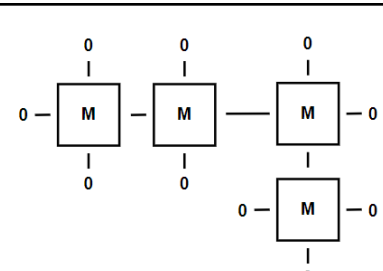
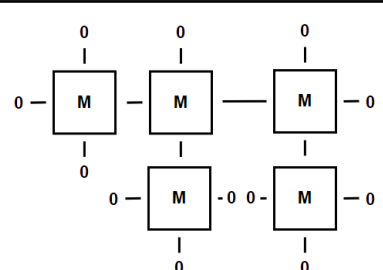
PJ - produkcja łącząca poziomo dwa sąsiednie elementy (join)

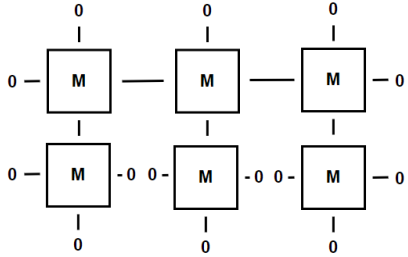
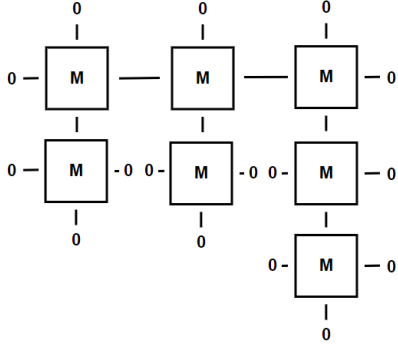
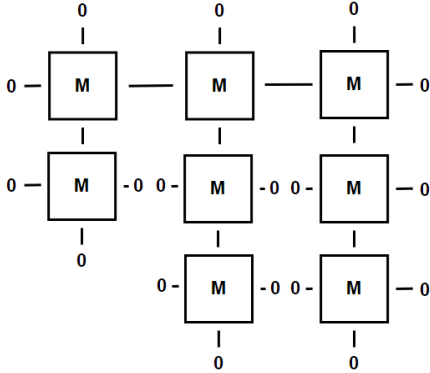
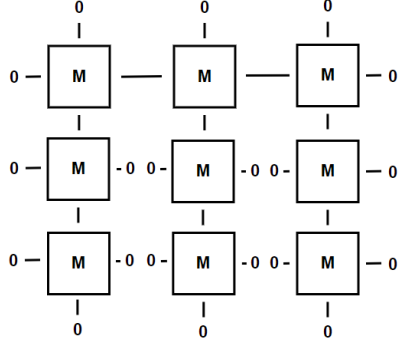


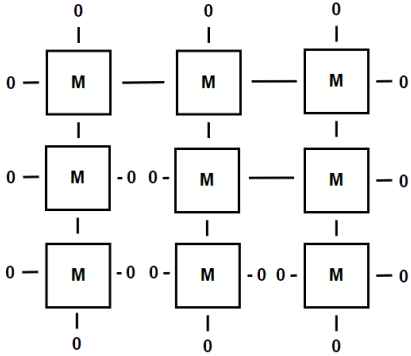
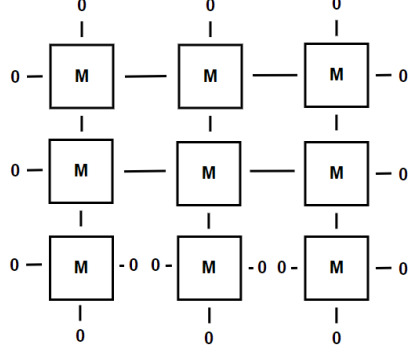
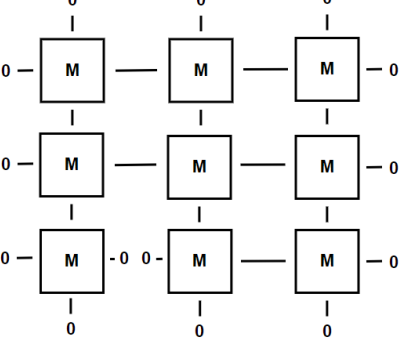
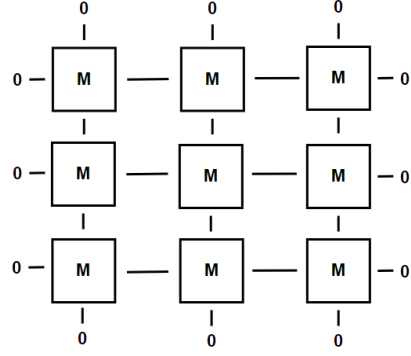
Możliwym ciągiem produkcji generującym siatkę 3x3 jest m.in. :

PI -> PW -> PW -> PS -> PS -> PS -> PS -> PS -> PJ -> PJ -> PJ -> PJ

Kolejne etapy generowania siatki, według powyższego ciągu produkcji

	S
PI	
PW PW ₁	
PW PW ₂	
PS PS _(1,0)	
PS PS _(1,1)	

<p>PS PS_(1,2)</p>	
<p>PS PS_(2,0)</p>	
<p>PS PS_(2,1)</p>	
<p>PS PS_(2,2)</p>	

<p>PJ</p> <p>$PJ_{(1,1)}$</p>	
<p>PJ</p> <p>$PJ_{(1,2)}$</p>	
<p>PJ</p> <p>$PJ_{(2,1)}$</p>	
<p>PJ</p> <p>$PJ_{(2,2)}$</p>	

Alfabet w sensie teorii śladów

$$A = \{PI, PW_1, PW_2, PS_{(1,0)}, PS_{(1,1)}, PS_{(1,2)}, PS_{(2,0)}, PS_{(2,1)}, PS_{(2,2)}, PJ_{(1,1)}, PJ_{(1,2)}, PJ_{(2,1)}, PJ_{(2,2)}\}$$

W ogólnym przypadku dla generacji siatki NxM alfabet to:

$$A = \{PI\} \cup \{PW_i \mid 0 < i < N\} \cup \{PS_{(i,j)} \mid 0 < i < N \text{ oraz } 0 \leq j < M\} \cup \{PJ_{(i,j)} \mid 0 < i < N \text{ oraz } 0 < j < M\}$$

kolumny liczone od prawej do lewej (0 do M-1), wiersze od góry w dół (0 do N-1)

PW_i - dołączenie elementu z lewej(West) strony w i-tej kolumnie

$PS_{(i,j)}$ - dołączenie elementu na dole(South) w i-tym wierszy oraz j-tej kolumnie

$PJ_{(i,j)}$ - połączenie elementu w i-tym wierszy oraz j-tej kolumnie z elementem po prawej(East)

Słowo odpowiadające generacji siatki

$$PI, PW_1, PW_2, PS_{(1,0)}, PS_{(1,1)}, PS_{(1,2)}, PS_{(2,0)}, PS_{(2,1)}, PS_{(2,2)}, PJ_{(1,1)}, PJ_{(1,2)}, PJ_{(2,1)}, PJ_{(2,2)}$$

Relacja zależności dla alfabetu A

$$D = \text{sym}\{ \{ (PI, PW_1), (PI, PS_{(1,0)}), (PW_1, PW_2), (PW_1, PS_{(1,1)}), (PW_2, PS_{(1,2)}), (PS_{(1,0)}, PS_{(2,0)}), (PS_{(1,0)}, PJ_{(1,1)}), (PS_{(1,1)}, PS_{(2,1)}), (PS_{(1,1)}, PJ_{(1,1)}), (PS_{(1,1)}, PJ_{(1,2)}), (PS_{(1,2)}, PS_{(2,2)}), (PS_{(1,2)}, PJ_{(1,2)}), (PS_{(2,0)}, PJ_{(2,1)}), (PS_{(2,1)}, PJ_{(2,1)}), (PS_{(2,1)}, PJ_{(2,2)}), (PS_{(2,2)}, PJ_{(2,2)}) \}^+ \} \cup I_A$$

Postać normalna Foaty

$$FNF = [PI][PW_1, PS_{(1,0)}][PW_2, PS_{(1,1)}, PS_{(2,0)}][PS_{(1,2)}, PS_{(2,1)}, PJ_{(1,1)}][PS_{(2,2)}, PJ_{(1,2)}, PJ_{(2,1)}][PJ_{(2,2)}]$$

Algorytm współbieżny dla siatki NxN

zaczynamy od PI,

po PI druga klasa wykonuje produkcje $PW_1, PS_{(1,0)}$,

jeśli klasa k, wykonuje PW_i to klasa k + 1 wykonuje $PW_{i+1}, (i+1 < N)$ oraz $PS_{(1,i)} i < N$,

jeśli klasa k, wykonuje $PS_{(i,j)}$ to klasa k + 1 wykonuje $PS_{(i+1,j)}, (i+1 < N)$

jeśli klasa k, wykonuje $PS_{(i,j)}$ oraz j jest różne od 0 to klasa k + 1 wykonuje $PJ_{(i,j)}$,

Wykorzystałem projekt pokazany na ćwiczeniach. Dodatkowe klasy, które stworzyłem na potrzeby zaimplementowania generacji siatki NxN to SquareElement, MatrixDrawer, PI, PW, PS, PJ, MatrixExecutor