**PDA TO CFG:**

Per convertire un PDA che accetta per stack vuoto in una CFG, si fanno questi passi:

Ogni variabile è una tripla [pXq] che indica la transizione dallo stato p allo stato q, eliminando X dallo stack. Quest'operazione può essere fatta anche in più passi.

1) Per ogni stato q del PDA, si aggiunge la produzione S → [p0 Z0 q]

dove p0 è lo stato iniziale del PDA

2) If δ (q, a, A) contains (q1, ε) then add

[q A q1] → a

dove a può essere anche epsilon, basta che in un'unica transizione si elimina il simbolo A dallo stack

3) If δ (q, a, A) contains (q1 , B1 B2 ...Bm ) then

• For every possible sequence of states q2 , ...qm+1

• Add

[q A qm+1] → a[q1 B1 q2] [q2 B2 q3] ... [qm Bm qm+1]

(a potrebbe essere anche epsilon, qua si guarda solo se da un simbolo se ne producono uno o più)

Se ci sono n stati, allora le produzioni di tipo 3 sono n^2.

Lo stato finale della testa e del corpo sono gli stessi (qm+1)

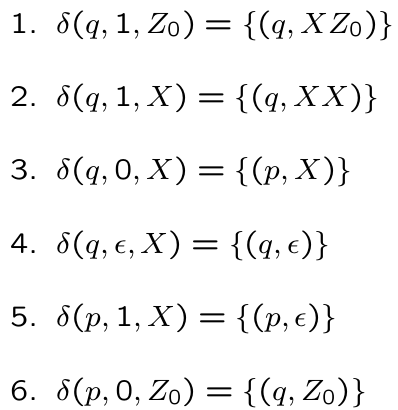
Gli stati intermedi del corpo sono gli stessi, la fine di uno è l'inizio del successivo, es:

… q2] [q2 … q3] [q3 … q4] ...

Per fare tutte le combinazioni basta fare la tabellina binaria.

**Esempio: Convertiamo P = ({p, q}, {0, 1}, {X, Z 0 }, δ, q, Z 0 ), dove δ**

**e’ data da:**

****

Per prima cosa, si usa la regola 1 di conversione, che produce le variabili a partire dal simbolo iniziale S:

Quindi, per ogni variabile del PDA, si mette la tripla (variabile della grammatica):

[q Z0 p]

dove q è lo stato iniziale del PDA e p è uno stato del PDA.

Quindi:

S → [q Z0 q]

S → [q Z0 p]

-------------------------------------------------------------------------------------------------------------------

Dalla regola 1 si ricavano queste produzioni usando la regola 3 di conversione:

[q Z0 x] → 1 [q X y] [y Z0 x]

dove x e y possono essere, rispettivamente:

|  |  |
| --- | --- |
| **x** | **y** |
|  |  |
| q | q |
| q | p |
| p | q |
| p | p |

Facendo tutte le combinazioni si ottiene quindi:

[q Z0 q] → 1 [q X q] [q Z0 q]

[q Z0 q] → 1 [q X p] [p Z0 q]

[q Z0 p] → 1 [q X q] [q Z0 p]

[q Z0 p] → 1 [q X p] [p Z0 p]

--------------------------------------------------------------------------------------------------------

Dalla regola 2 si ottengono queste produzioni usando la regola 3 di conversione:

[q X x] → 1 [q X y] [y X x]

dove x e y possono essere, rispettivamente:

|  |  |
| --- | --- |
| **x** | **y** |
|  |  |
| q | q |
| q | p |
| p | q |
| p | p |

Quindi:

[q X q] → 1 [q X q] [q X q]

[q X q] → 1 [q X p] [p X q]

[q X p] → 1 [q X q] [q X p]

[q X p] → 1 [q X p] [p X p]

---------------------------------------------------------------------------

Dalla regola 3 si ottengono queste produzioni usando sempre la regolae 3 di conversione:

[q X x] → 0 [p X x]

dove x può essere:

|  |
| --- |
| **x** |
|  |
| q |
| p |

Quindi si ottiene:

[q X q] → 0 [p X q]

[q X p] → 0 [p X p]

-----------------------------------------------------------------------------------------------------

Dalla regola 4 si ottengono queste produzioni usando la regola 2 di conversione:

[q X q] → epsilon

----------------------------------------------------------------------------------------------------

Dalla regola 5 di conversione si ottengono queste produzioni usando la regola 2 di conversione:

[p X p] → 1

--------------------------------------------------------------------------------------------------

Dalla regola 6 di conversione si ottengono queste produzioni usando la regola 3 di conversione:

[p Z0 x] → 0 [q Z0 x]

dove x può essere:

|  |
| --- |
| **x** |
|  |
| q |
| p |

Quindi:

[p Z0 q] → 0 [q Z0 q]

[p Z0 p] → 0 [q Z0 p]

-------------------------------------------------------------------------------------  
Si possono infine raggruppare tutte le produzioni per ogni variabile, e così si ottiene la grammatica finale.