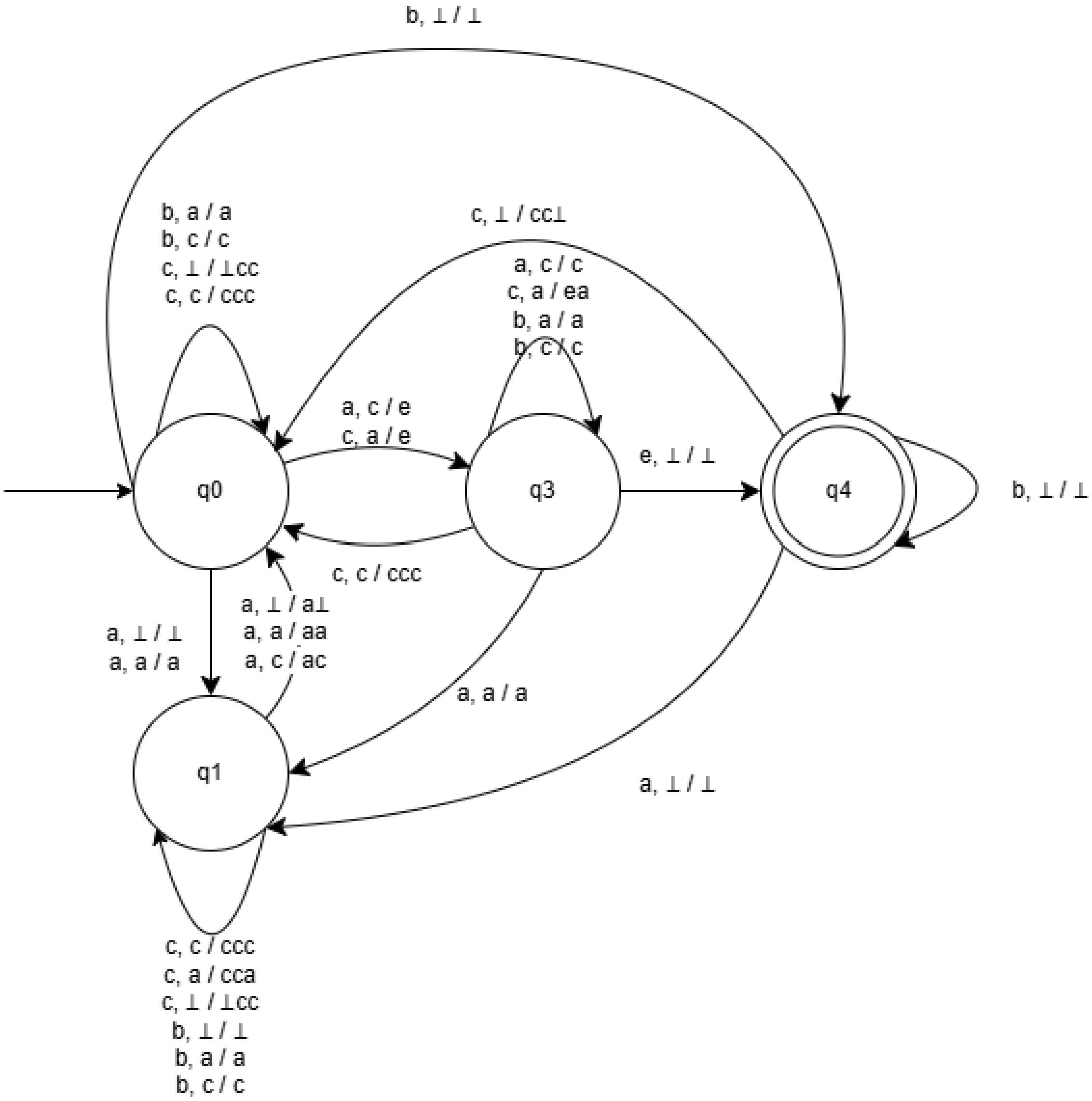
A2.1:

DPDA:



Wort1 = bcaba

Start: q0

1. q0 -> b -> q4 Stack: ⊥
2. q4 -> c -> q0 Stack: cc⊥
3. q0 -> a -> q3 Stack: c⊥
4. q3 -> b -> q3 Stack: c⊥
5. q3 -> a -> q3 Stack: ⊥
6. q3 -> e -> q4 Stack: ⊥ -> Wort akzeptiert

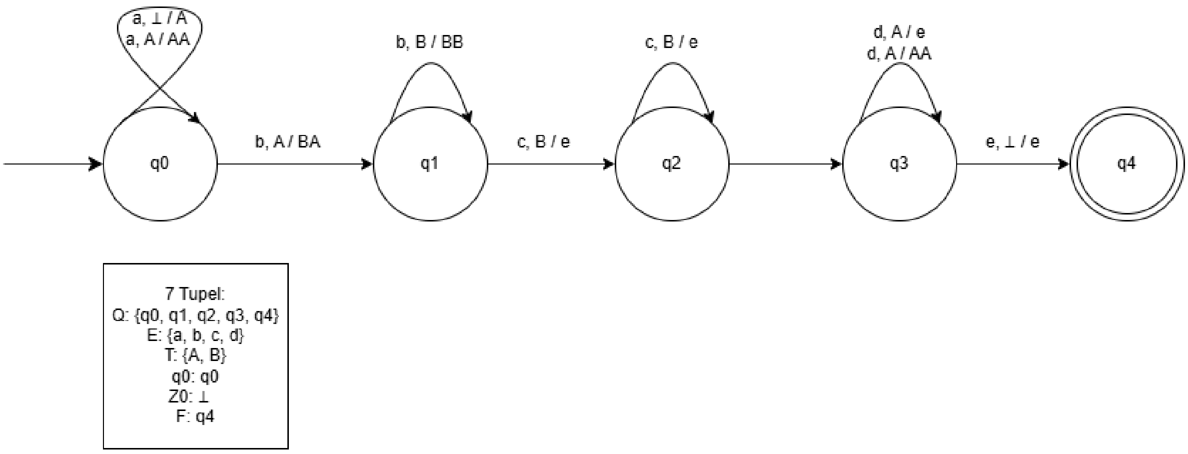
Wort2 = bccac

Start: q0

1. q0 -> b -> q4 Stack: ⊥
2. q4 -> c -> q0 Stack: cc⊥
3. q0 -> c -> q0 Stack: cccc⊥
4. q0 -> a -> q3 Stack: ccc⊥
5. q3 -> c -> q0 Stack: ccccc⊥ -> Wort nicht akzeptiert

A2.2:

PDA:



Sprache: L = $\{a^n b^m c^m d^n\}$

A2.3:

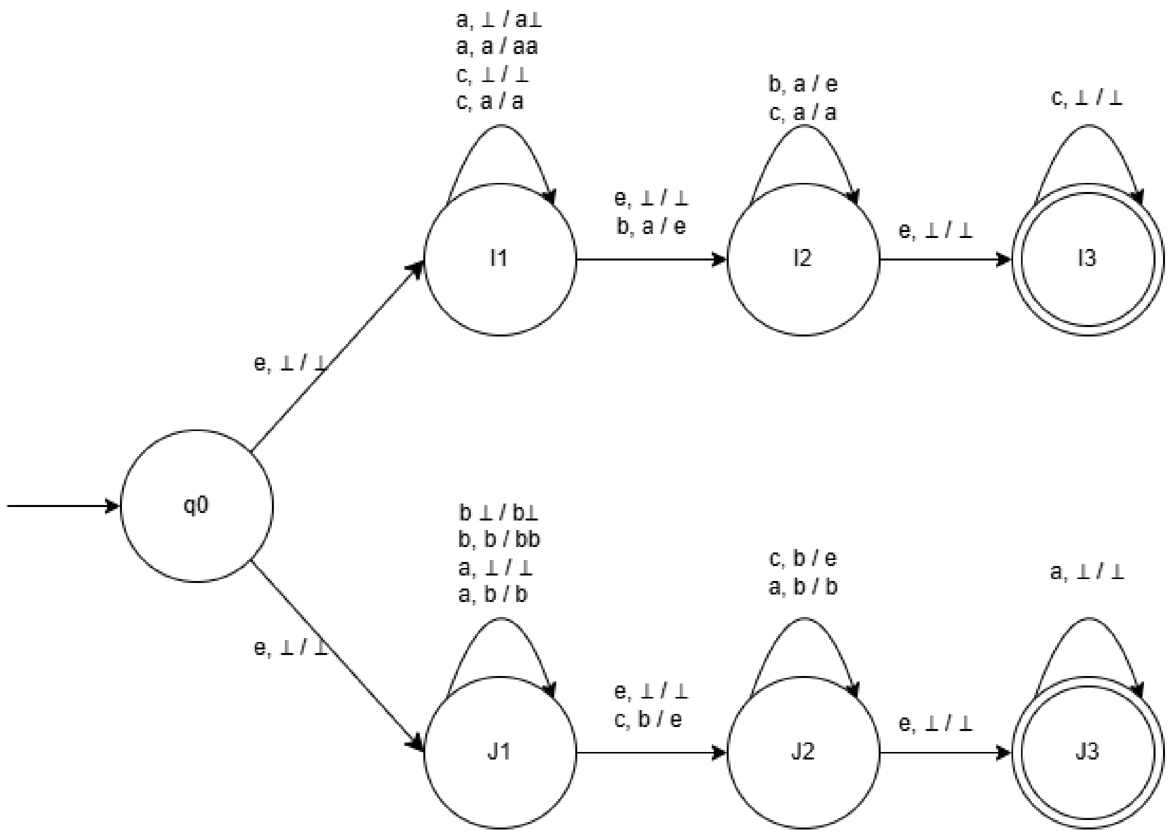
Condition = C, Statement = S

L(g) = ({ifCS(elseS)?)+} es wird eine verschachtelte Konstruktion aus if und else Statements generiert.

Diese Grammatik ist mehrdeutig da man dasselbe Wort unterschiedlich Interpretieren kann, also das man abhängig, wo man eine Klammer setzt, unterschiedlich Ableitungsbäume bilden kann.

A2.4:

PDA:



L = {a^i b^j c^k | i = j v j = k}

G = ({q0, I, J, A1, C1, C2, A2}, {a, b, c}, P, q0})

q0 -> I | J

I -> A1 C1

A1 -> a A1 b | ab

C1 -> cC1 | c

J -> C2 A2

C2 -> c C1 b | cb

A2 -> aA1 | a

Diese Grammatik ist mehrdeutig da, wenn i = j = k dann können zwei Ableitungsbäume erstellt werden bei dem einer von q0 nach I springt und der andere von q0 zu J, beide aber das gleiche Wort akzeptieren. Zum Beispiel bei w=(aabbcc).