Equivalence of NPDAs and CFGs

CFLs $\equiv$ Languages Generated by CFGs

CFGs can be mapped to equivalent NPDAs. Implies CFGs $\subseteq$ languages accepted by NPDAs.

Idea of mapping:
Map each production rule to a transition.

NPDAs can be mapped into equivalent CFGs. This implies NPDAs $\subseteq$ CFGs.

Idea of mapping:
For each $q_i, q_j \in Q$ and $A \in \Gamma$,
create a variable symbol $\langle q_i, A, q_j \rangle$.

Map each transition to production rules.

NPDAs $\subseteq$ CFGs $\subseteq$ NPDAs
implies NPDAs $=\equiv$ CFGs.
A transition from $q_0$ to $q_1$ pushes $S$ on the stack.

Each production rule $A \rightarrow u$ goes to a $q_1, q_1$ transition that pops $A$ and pushes $u$.

Each terminal symbol $a$ goes to a $q_1, q_1$ transition that reads $a$ and pops $a$.

A transition from $q_1$ to $q_f$ is allowed when the stack is down to $z$. 
Transforming NPDAs to CFGs

Assume one final state $q_f$ which is entered iff the stack is empty. Start symbol is $\langle q_0, z, q_f \rangle$.

$\langle q_i, A, q_j \rangle \Rightarrow^* w$ iff $(q_i, w, A) \vdash^* (q_j, \lambda, \lambda)$, i.e., the NPDA can read $w$ and “erase” $A$ while going from $q_i$ to $q_j$.

Each transition $(q_j, \lambda) \in \delta(q_i, a, A)$ is transformed to a production rule $\langle q_i, A, q_j \rangle \rightarrow a$.

$(q_j, B) \in \delta(q_i, a, A)$ goes to:

$\langle q_i, A, q_k \rangle \rightarrow a\langle q_j, B, q_k \rangle$ for all $q_k \in Q$

$(q_j, BC) \in \delta(q_i, a, A)$ goes to:

$\langle q_i, A, q_l \rangle \rightarrow a\langle q_j, B, q_k \rangle\langle q_k, C, q_l \rangle$

for all $q_k, q_l \in Q$. 