

Some notes
on the equivalence of npda's and context-free grammars.

Theorem. *A language is context-free iff there is an npda that accepts it.*

This theorem is proved in two Lemmas:

Lemma 1. *Let G be a context-free grammar. Then there is an npda M such that $L(M) = L(G)$.*

Lemma 2. *Let M be an npda. Then there is a context-free grammar G such that $L(M) = L(G)$.*

For the proof of Lemma 1, we will describe a procedure which, given a context-free grammar $G = (V, T, S, P)$, produces an equivalent npda $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$. We set $Q = \{q_0, q_1, q_f\}$, $F = \{q_f\}$, $\Sigma = T$, $\Gamma = V \cup T \cup \{z\}$. Now for the δ function:

- (1) $\delta(q_0, \lambda, z) = \{(q_1, Sz)\}$.
- (2) $(q_1, w) \in \delta(q_1, \lambda, A)$ for all $w \in (V \cup T)^*$ and all $A \in V$ such that $A \rightarrow w$ is in P .
- (3) $\delta(q_1, a, a) = \{(q_1, \lambda)\}$ for each $a \in \Sigma$.
- (4) $\delta(q_1, \lambda, z) = \{(q_f, \lambda)\}$.

Example. Let G be the grammar given by:

$$\begin{aligned} S &\rightarrow aSb \mid A \mid B \mid \lambda \\ A &\rightarrow aA \mid \lambda \\ B &\rightarrow bB \mid bb \end{aligned}$$

This grammar is my solution to exercise (6b) from section 5.1. $L(G) = \{a^n b^m : n \neq m - 1\}$. Let us employ the procedure described above. This will give us an npda M with the following δ function.

$$\begin{aligned} \delta(q_0, \lambda, z) &= \{(q_1, Sz)\} \\ \delta(q_1, \lambda, S) &= \{(q_1, aSb), (q_1, A), (q_1, B), (q_1, \lambda)\} \\ \delta(q_1, \lambda, A) &= \{(q_1, aA), (q_1, \lambda)\} \\ \delta(q_1, \lambda, B) &= \{(q_1, bB), (q_1, bb)\} \\ \delta(q_1, a, a) &= \{(q_1, \lambda)\} \\ \delta(q_1, b, b) &= \{(q_1, \lambda)\} \\ \delta(q_1, \lambda, z) &= \{(q_f, \lambda)\} \end{aligned}$$

In order to see that the npda M is equivalent to the grammar G , notice that there is a one-to-one correspondence between left-most derivations of sentences from G and sequences of valid moves of M .

Example. Here is a typical derivation from G :

$$S \Rightarrow asb \Rightarrow aaSbb \Rightarrow aaAbb \Rightarrow aaaAbb \Rightarrow aaabb.$$

Here is the corresponding sequence of valid moves of M :

$$\begin{aligned} (q_0, aaabb, z) \vdash & (q_1, aaabb, Sz) \vdash (q_1, aaabb, aSbz) \vdash (q_1, aabb, Sbz) \vdash (q_1, aabb, aSbbz) \vdash (q_1, abb, Sbbz) \vdash \\ & (q_1, abb, Abbz) \vdash (q_1, abb, aAbbz) \vdash (q_1, bb, Abbz) \vdash (q_1, bb, bbz) \vdash (q_1, b, bz) \vdash (q_1, \lambda, z) \vdash (q_1, \lambda, \lambda) \vdash \\ & \text{“accept”} \end{aligned}$$

That is all I will say about Lemma 1. Now let us discuss Lemma 2.

Definition. Let $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ be an npda. We say that M is in *standard reduced form* iff

- (a) M has exactly three states: $Q = \{q_0, q_1, q_f\}$.
- (b) $F = \{q_f\}$.
- (c) There is a symbol $S \in \Gamma$ with $S \neq z$ such that the only δ clause which involves the state q_0 is the clause: $\delta(q_0, \lambda, z) = \{(q_1, Sz)\}$.
- (d) The only δ -clause which involves the state q_f is the clause: $\delta(q_1, \lambda, z) = \{(q_f, \lambda)\}$.

Notice that if G is a context-free grammar and M is the equivalent npda which is produced by the procedure described on the previous page, then M is in standard reduced form. Well now if M' is *any* npda in standard reduced form we can more-or-less invert the previous procedure in order to produce an equivalent context-free grammar G' .

Sub-Lemma 2a. *Let M be an npda in standard reduced form. Then there is a context-free grammar G such that $L(M) = L(G)$.*

For the proof of Sub-Lemma 2a, we will now describe a procedure which, given an npda $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ in standard reduced form, produces an equivalent context-free grammar $G = (V, T, S, P)$. We set $V = \Gamma$, $T = \Sigma$, S = the S mentioned in clause (c) of the definition of standard reduced form. Now for the definition of P . P will include the rule $A \rightarrow aw$ for every $A \in \Gamma$ and every $a \in \Sigma \cup \{\lambda\}$ and every $w \in \Gamma^*$ such that $(q_1, w) \in \delta(q_1, a, A)$.

Example. Let M be the npda given by the following. $\Gamma = \{S, A, B, C, z\}$. $F = \{q_f\}$.

$$\begin{aligned} \delta(q_0, \lambda, z) &= \{(q_1, Sz)\}. \\ \delta(q_1, \lambda, S) &= \{(q_1, A), (q_1, B), (q_1, \lambda)\}. \\ \delta(q_1, a, S) &= \{(q_1, SC)\}. \\ \delta(q_1, b, C) &= \{(q_1, \lambda)\}. \\ \delta(q_1, \lambda, A) &= \{(q_1, \lambda)\}. \\ \delta(q_1, a, A) &= \{(q_1, A)\}. \\ \delta(q_1, \lambda, B) &= \{(q_1, bb)\}. \\ \delta(q_1, b, B) &= \{(q_1, B)\}. \\ \delta(q_1, \lambda, z) &= \{(q_f, \lambda)\}. \end{aligned}$$

First notice that M is in fact in standard reduced form. Now we apply the procedure to produce an equivalent context-free grammar G . G is given by the following:

$$\begin{aligned} S &\rightarrow A \mid B \mid \lambda \mid aSC \\ C &\rightarrow b \\ A &\rightarrow \lambda \mid aA \\ B &\rightarrow bb \mid bB \end{aligned}$$

So to complete the proof of Lemma 2 we now need to prove:

Sub-Lemma 2b. *Let M be any npda. Then there is an equivalent npda in standard reduced form.*

Definition. Let $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ be an npda. We say that M is in *pre-standard form* iff

- (i) M has a single final state: $F = \{q_f\}$, and $q_f \neq q_0$.
- (ii) The only δ -clauses which involve the state q_f are clauses of the form: $\delta(q, \lambda, z) = \{(q_f, \lambda)\}$, where q is some state other than q_f .

We now discuss a procedure for converting any npda into an equivalent one in standard reduced form. Let $M' = (Q', \Sigma', \Gamma', \delta', q'_0, z', F')$ be any npda.

Step One. First we will convert M' into an equivalent npda $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ such that M is in pre-standard form. To carry out Step One we define $Q = Q' \cup \{q_0, q_f, q^*\}$. (Assume Q' does not contain any elements called q_0, q_f or q^* .) We define $\Gamma = \Gamma' \cup \{z\}$. (Assume Γ' does not have any elements called z .) We define $\Sigma = \Sigma'$ and $F = \{q_f\}$. Now for the δ function:

$$\delta(q_0, \lambda, z) = \{(q'_0, z'z)\}.$$

δ contains all of the clauses from δ' .

$$\delta(q, \lambda, x) = \{(q^*, x)\}, \text{ for each } q \in F' \text{ and each } x \in \Gamma.$$

$$\delta(q^*, \lambda, x) = \{(q^*, \lambda)\}, \text{ for each } x \in \Gamma'.$$

$$\delta(q^*, \lambda, z) = \{(q_f, \lambda)\}.$$

Example. Suppose M' is given by the following. $Q' = \{q'_0, q'_1, q'_2, q'_3\}$, $F' = \{q'_2, q'_3\}$, $\Sigma' = \{a\}$, $\Gamma' = \{z', 1\}$, and δ' is given by:

$$\delta'(q'_0, a, z') = \{(q'_1, 1z'), (q'_2, \lambda)\}$$

$$\delta'(q'_1, a, 1) = \{(q'_3, 11)\}.$$

Notice that $L(M') = \{a, aa\}$. Here are the two sequences of instantaneous descriptions showing this:

$$(q'_0, a, z') \vdash (q'_2, \lambda, \lambda) \vdash \text{“accept”}$$

$$(q'_0, aa, z') \vdash (q'_1, a, 1z') \vdash (q'_3, \lambda, 11z') \vdash \text{“accept”}$$

M' is not in pre-standard form. Notice that M' has two final states and also that in the second sequence above, the stack is not empty at the end. Now we carry out Step 1 described above. We will convert M' into an equivalent npda M such that M is in pre-standard form. For M we will have $Q = \{q_0, q'_0, q'_1, q'_2, q'_3, q^*, q_f\}$, $F = \{q_f\}$, $\Sigma = \{a\}$, $\Gamma = \{z, z', 1\}$. See the next page for the δ function.

$$\delta(q_0, \lambda, z) = \{(q'_0, z'z)\}.$$

$$\begin{aligned} \delta(q'_0, a, z') &= \{(q'_1, 1z'), (q'_2, \lambda)\} \\ \delta(q'_1, a, 1) &= \{(q'_3, 11)\}. \end{aligned}$$

$$\begin{aligned} \delta(q'_2, \lambda, 1) &= \{(q^*, 1)\} \\ \delta(q'_2, \lambda, z') &= \{(q^*, z')\} \\ \delta(q'_2, \lambda, z) &= \{(q^*, z)\} \\ \delta(q'_3, \lambda, 1) &= \{(q^*, 1)\} \\ \delta(q'_3, \lambda, z') &= \{(q^*, z')\} \\ \delta(q'_3, \lambda, z) &= \{(q^*, z)\} \end{aligned}$$

$$\delta(q^*, \lambda, z) = \{(q_f, \lambda)\}$$

Notice that $L(M) = L(M') = \{a, aa\}$. Here are the two sequences of instantaneous descriptions of M that correspond to the two sequences of M' given on the previous page:

$$(q_0, a, z) \vdash (q'_0, a, z'z) \vdash (q'_2, \lambda, z) \vdash (q^*, \lambda, z) \vdash (q_f, \lambda, \lambda) \vdash \text{“accept”}$$

$$(q_0, aa, z) \vdash (q'_0, aa, z'z) \vdash (q'_1, a, 1z'z) \vdash (q'_3, \lambda, 11z'z) \vdash (q^*, \lambda, 11z'z) \vdash (q^*, \lambda, 1z'z) \vdash (q^*, \lambda, z'z) \vdash (q^*, \lambda, z) \vdash (q_f, \lambda, \lambda) \vdash \text{“accept”}$$

Notice that M is in pre-standard form.

Step 2. Now suppose we are given an npda $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ such that M is an pre-standard form. So there is some $q_f \in Q$ with $q_f \neq q_0$ such that $F = \{q_f\}$.

We will now describe a procedure for constructing an equivalent npda $\bar{M} = (\bar{Q}, \bar{\Sigma}, \bar{\Gamma}, \bar{\delta}, \bar{q}_0, \bar{z}, \bar{F})$ such that \bar{M} is in standard reduced form. We set $\bar{Q} = \{\bar{q}_0, \bar{q}_1, \bar{q}_f\}$, $\bar{F} = \{\bar{q}_f\}$, $\bar{\Sigma} = \Sigma$. We also set $\bar{\Gamma} = (Q \times \Gamma \times Q) \cup \{\bar{z}\}$. That is, in addition to the start-stack symbol \bar{z} , the other members of our stack alphabet will all be triples of the form (q_i, A, q_j) where q_i and q_j are states of M , and A is a symbol from the stack alphabet of M . Since \bar{M} is going to be in standard-reduced form, we must choose one of the stack symbols to be \bar{S} . We will use $\bar{S} = (q_0, z, q_f)$.

The idea here is that \bar{M} is going to simulate M . Since \bar{M} has only one working state, namely \bar{q}_1 , whereas M may have many states, \bar{M} needs to keep track of what state M is in by recording the information using the stack symbols. If (q_i, A, q_j) is on the top of the stack in \bar{M} , this will represent the situation that M is in state q_i and M has an A on the top of the stack. The meaning of the q_j is a little more obscure. Roughly speaking it means that after M pops the A off the stack then M will be in state q_j .

Now we define $\bar{\delta}$. Since \bar{M} will be in standard-reduced form we know that we will have at least the following two clauses:

$$\bar{\delta}(\bar{q}_0, \lambda, \bar{z}) = \{(\bar{q}_1, \bar{S}\bar{z})\} \quad \bar{\delta}(\bar{q}_1, \lambda, \bar{z}) = \{(\bar{q}_f, \lambda)\}$$

That is, the very first thing \bar{M} does is push \bar{S} on top of \bar{z} and then go to \bar{q}_1 . (Remember that $\bar{S} = (q_0, z, q_f)$.) Then \bar{M} will do all of its calculations in state \bar{q}_1 , until the \bar{z} again appears on the top of the stack. At this point \bar{M} will pop \bar{z} leaving the stack empty, go to state \bar{q}_f , and then quit.

Now for the rest of $\bar{\delta}$.

$$(\bar{q}_1, \lambda) \in \bar{\delta}(\bar{q}_1, a, (q_i, A, q_j)) \text{ whenever } q_i, q_j \in Q, a \in \Sigma \cup \{\lambda\}, A \in \Gamma, \text{ and } (q_j, \lambda) \in \delta(q_i, a, A).$$

That is, if M can read a , pop A , and go from q_i to q_j , then \bar{M} can read a , pop (q_i, A, q_j) , and remain in state \bar{q}_1 .

There is one last part to the definition of $\bar{\delta}$. The previous clauses show us how \bar{M} *decreases* the size of its stack in order to simulate M decreasing the size of its stack. We now need to describe how \bar{M} *increases* the size of its stack in order to simulate M increasing the size of its stack.

The following discussion gets a bit complicated because of the notation. If you have trouble following it, try reading the example that comes next.

$(\bar{q}_1, (q_j, A_1, q_{x1})(q_{x1}, A_2, q_{x2}) \cdots (q_{x(n-1)}, A_n, q_k)) \in \bar{\delta}(\bar{q}_1, a, (q_i, A, q_k))$ whenever q_i, q_j, q_k , and $q_{x1}, \dots, q_{x(n-1)}$ are in Q ; A, A_1, \dots, A_n are in Γ ; $a \in \Sigma \cup \{\lambda\}$; and $(q_j, A_1 A_2 \cdots A_n) \in \delta(q_i, a, A)$.

In other words, whenever M can read a , pop A , push $A_1 A_2 \cdots A_n$, and go from state q_i to state q_j , then \bar{M} can read a , pop (q_i, A, q_k) for any q_k , push $(q_j, A_1, q_{x1})(q_{x1}, A_2, q_{x2}) \cdots (q_{x(n-1)}, A_n, q_k)$ for any sequence $q_{x1}, \dots, q_{x(n-1)}$, and remain in state \bar{q}_1 .

Notice that the right-most component of the right-most stack symbol being pushed in \bar{M} : the q_k in $(q_{x(n-1)}, A_n, q_k)$, must be the same as the right-most component of the stack symbol being popped: the q_k in (q_i, A, q_k) . Also notice that the left-most component of the left-most stack symbol being pushed: the q_j in (q_j, A_1, q_{x1}) , must be precisely the new state to which M is going. But the other components: $q_{x1}, q_{x2}, \dots, q_{x(n-1)}$, can be any sequence of elements from Q .

Note also that the components are organized so the the right-most component of each stack symbol being pushed in \bar{M} is the same as the left-most component of the stack symbol just below it on the stack. That is, the q_{x1} in (q_j, A_1, q_{x1}) is the same as the q_{x1} in (q_{x1}, A_2, q_{x2}) . This makes sense because, as I mentioned before, the q_{x1} in (q_j, A_1, q_{x1}) represents, roughly speaking, the fact that if M were to pop A_1 off of the top of its stack then M would be in state q_{x1} . But if M were to pop A_1 off of the top of its stack, then in the simulation of M by \bar{M} , \bar{M} would pop (q_i, A_1, q_{x1}) off the top of its stack. This would leave (q_{x1}, A_2, q_{x2}) on the top of \bar{M} 's stack, and as I mentioned before, the q_{x1} in (q_{x1}, A_2, q_{x2}) represents the fact that M is in state q_{x1} .

Example. Let M be the npda given by the following. $Q = \{q_0, q_1, q_f\}$, $F = \{q_f\}$, $\Sigma = \{a, b\}$, $\Gamma = \{1, z\}$, and δ is given by:

$$\begin{aligned} \delta(q_0, \lambda, z) &= \{(q_f, \lambda)\} \\ \delta(q_0, a, z) &= \{(q_0, 11z)\} \\ \delta(q_0, a, 1) &= \{(q_0, 111)\} \\ \delta(q_0, b, 1) &= \{(q_1, \lambda)\} \\ \delta(q_1, b, 1) &= \{(q_1, \lambda)\} \\ \delta(q_1, \lambda, z) &= \{(q_f, \lambda)\} \end{aligned}$$

Notice that $L(M) = \{a^n b^{2n} : n \geq 0\}$. Also note that M is in pre-standard form but not in standard reduced form, because M actually does real work in state q_0 .

We will now perform Step 2 on M to produce an equivalent npda $\bar{M} = (\bar{Q}, \bar{\Sigma}, \bar{\Gamma}, \bar{\delta}, \bar{q}_0, \bar{z}, \bar{F})$ such that \bar{M} is in standard reduced form. We set $\bar{Q} = \{\bar{q}_0, \bar{q}_1, \bar{q}_f\}$, $\bar{F} = \{\bar{q}_f\}$, $\bar{\Sigma} = \{a, b\}$. We also set $\bar{\Gamma} = (\{q_0, q_1, q_f\} \times \{1, z\} \times \{q_0, q_1, q_f\}) \cup \{\bar{z}\}$. Since \bar{M} is going to be in standard-reduced form, we must choose one of the stack symbols to be \bar{S} . We will use $\bar{S} = (q_0, z, q_f)$.

Now we define $\bar{\delta}$. Since \bar{M} will be in standard-reduced form we have the following two clauses:

$$(1) \bar{\delta}(\bar{q}_0, \lambda, \bar{z}) = \{(\bar{q}_1, \bar{S}\bar{z})\} \quad (2) \bar{\delta}(\bar{q}_1, \lambda, \bar{z}) = \{(\bar{q}_f, \lambda)\}.$$

Now for the rest of $\bar{\delta}$. First the clauses that define when \bar{M} can *decrease* the size of its stack:

<u>These clauses for \bar{M}</u>	<u>come from these clauses for M.</u>
(3) $\bar{\delta}(\bar{q}_1, \lambda, (q_0, z, q_f)) = \{(\bar{q}_1, \lambda)\}$	$\delta(q_0, \lambda, z) = \{(q_f, \lambda)\}$
(4) $\bar{\delta}(\bar{q}_1, b, (q_0, 1, q_1)) = \{(\bar{q}_1, \lambda)\}$	$\delta(q_0, b, 1) = \{(q_1, \lambda)\}$
(5) $\bar{\delta}(\bar{q}_1, b, (q_1, 1, q_1)) = \{(\bar{q}_1, \lambda)\}$	$\delta(q_1, b, 1) = \{(q_1, \lambda)\}$
(6) $\bar{\delta}(\bar{q}_1, \lambda, (q_1, z, q_f)) = \{(\bar{q}_1, \lambda)\}$	$\delta(q_1, \lambda, z) = \{(q_f, \lambda)\}$

Next the clauses that define when \bar{M} can *increase* the size of its stack.

In M we have the clause: $\delta(q_0, a, z) = \{(q_0, 11z)\}$. This yields the following *clause template* in \bar{M} :

$$(7) \quad \bar{\delta}(\bar{q}_1, a, (q_0, z, q_x)) = \left\{ \left(\bar{q}_1, (q_0, 1, q_y)(q_y, 1, q_z)(q_z, z, q_x) \right) \right\}.$$

Notice that the above template is actually a short-hand for many different clauses. If we plug in any values for q_x , q_y , and q_z from the set $Q = \{q_0, q_1, q_f\}$, we will get a $\bar{\delta}$ -clause. So one δ -clause from M corresponds to many different $\bar{\delta}$ -clauses from \bar{M} . This clause template notation is very useful because it would take a lot of time and space to write out all of the individual clauses.

We now explain the method used to form the clause template: the a in $\bar{\delta}(\bar{q}_1, a, (q_0, z, q_x))$ comes from the a in $\delta(q_0, a, z)$. The q_0 in $\bar{\delta}(\bar{q}_1, a, (q_0, z, q_x))$ comes from the q_0 in $\delta(q_0, a, z)$. The z in $\bar{\delta}(\bar{q}_1, a, (q_0, z, q_x))$ comes from the z in $\delta(q_0, a, z)$. The q_0 in $(q_0, 1, q_y)(q_y, 1, q_z)(q_z, z, q_x)$ comes from the q_0 in $(q_0, 11z)$. The $11z$ as the middle components of $(q_0, 1, q_y)(q_y, 1, q_z)(q_z, z, q_x)$ comes from the $11z$ in $(q_0, 11z)$.

In general we will just use the above *clause template* notation and not bother writing down all of the actual $\bar{\delta}$ -clauses that the clause template represents. But just for this time, since it is the first time you are seeing this, I will actually write out all of the $\bar{\delta}$ -clauses. The above clause template expands into the following three huge clauses:

$$(7a) \quad \bar{\delta}(\bar{q}_1, a, (q_0, z, q_0)) = \left\{ \begin{array}{ll} \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_0)(q_0, z, q_0) \right), & \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_1)(q_1, z, q_0) \right), \\ \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_f)(q_f, z, q_0) \right), & \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_0)(q_0, z, q_0) \right), \\ \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_0) \right), & \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_f)(q_f, z, q_0) \right), \\ \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_0)(q_0, z, q_0) \right), & \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_1)(q_1, z, q_0) \right), \\ \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_f)(q_f, z, q_0) \right) \end{array} \right\}$$

$$(7b) \quad \bar{\delta}(\bar{q}_1, a, (q_0, z, q_1)) = \left\{ \begin{array}{ll} \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_0)(q_0, z, q_1) \right), & \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_1)(q_1, z, q_1) \right), \\ \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_f)(q_f, z, q_1) \right), & \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_0)(q_0, z, q_1) \right), \\ \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_1) \right), & \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_f)(q_f, z, q_1) \right), \\ \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_0)(q_0, z, q_1) \right), & \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_1)(q_1, z, q_1) \right), \\ \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_f)(q_f, z, q_1) \right) \end{array} \right\}$$

$$(7c) \quad \bar{\delta}(\bar{q}_1, a, (q_0, z, q_f)) = \left\{ \begin{array}{ll} \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_0)(q_0, z, q_f) \right), & \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_1)(q_1, z, q_f) \right), \\ \left(\bar{q}_1, (q_0, 1, q_0)(q_0, 1, q_f)(q_f, z, q_f) \right), & \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_0)(q_0, z, q_f) \right), \\ \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_f) \right), & \left(\bar{q}_1, (q_0, 1, q_1)(q_1, 1, q_f)(q_f, z, q_f) \right), \\ \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_0)(q_0, z, q_f) \right), & \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_1)(q_1, z, q_f) \right), \\ \left(\bar{q}_1, (q_0, 1, q_f)(q_f, 1, q_f)(q_f, z, q_f) \right) \end{array} \right\}$$

Finally there is one more set of clauses which increase the size of the stack. In M we have the clause: $\delta(q_0, a, 1) = \{(q_0, 111)\}$. This yields the following clause template in \bar{M} :

$$(8) \quad \bar{\delta}(\bar{q}_1, a, (q_0, 1, q_x)) = \left\{ \left(\bar{q}_1, (q_0, 1, q_y)(q_y, 1, q_z)(q_z, 1, q_x) \right) \right\}.$$

Again we explain the method used to form the clause template: the a in $\bar{\delta}(\bar{q}_1, a, (q_0, 1, q_x))$ comes from the a in $\delta(q_0, a, 1)$. The q_0 in $\bar{\delta}(\bar{q}_1, a, (q_0, 1, q_x))$ comes from the q_0 in $\delta(q_0, a, 1)$. The 1 in $\bar{\delta}(\bar{q}_1, a, (q_0, 1, q_x))$ comes from the 1 in $\delta(q_0, a, 1)$. The q_0 in $(q_0, 1, q_y)(q_y, 1, q_z)(q_z, 1, q_x)$ comes from the q_0 in $(q_0, 111)$. The 111 as the middle components of $(q_0, 1, q_y)(q_y, 1, q_z)(q_z, 1, q_x)$ comes from the 111 in $(q_0, 111)$.

In order to better understand our construction of \bar{M} , let us look at a few sequences of instantaneous descriptions of M which lead to an acceptance of a string, and then the corresponding sequence of instantaneous descriptions of \bar{M} .

Example. Let us look at accepting the string: λ .

First in M :

$$(q_0, \lambda, z) \vdash (q_f, \lambda, \lambda) \vdash \text{“accept”}$$

Next in \bar{M} .

$$(\bar{q}_0, \lambda, \bar{z}) \vdash (\bar{q}_1, \lambda, (q_0, z, q_f)\bar{z}) \vdash (\bar{q}_1, \lambda, \bar{z}) \vdash (\bar{q}_f, \lambda, \lambda) \vdash \text{“accept”}$$

Here is an explanation of each of the above moves: $(\bar{q}_0, \lambda, \bar{z}) \vdash (\bar{q}_1, \lambda, (q_0, z, q_f)\bar{z})$ uses clause (1) on page 5. Recall that $\bar{S} = (q_0, z, q_f)$. $(\bar{q}_1, \lambda, (q_0, z, q_f)\bar{z}) \vdash (\bar{q}_1, \lambda, \bar{z})$ uses clause (3) on page 6. $(\bar{q}_1, \lambda, \bar{z}) \vdash (\bar{q}_f, \lambda, \lambda)$ uses clause (2) on page 5.

Example. Next let us look at accepting the string: abb .

First in M :

$(q_0, abb, z) \vdash (q_0, bb, 11z) \vdash (q_1, b, 1z) \vdash (q_1, \lambda, z) \vdash (q_f, \lambda, \lambda) \vdash$ “**accept**”

Next in \bar{M} .

$(\bar{q}_0, abb, \bar{z}) \vdash (\bar{q}_1, abb, (q_0, z, q_f)\bar{z}) \vdash (\bar{q}_1, bb, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_f)\bar{z}) \vdash (\bar{q}_1, b, (q_1, 1, q_1)(q_1, z, q_f)\bar{z}) \vdash (\bar{q}_1, \lambda, (q_1, z, q_f)\bar{z}) \vdash (\bar{q}_1, \lambda, \bar{z}) \vdash (\bar{q}_f, \lambda, \lambda) \vdash$ “**accept**”

Here is an explanation of each of the above moves: $(\bar{q}_0, abb, \bar{z}) \vdash (\bar{q}_1, abb, (q_0, z, q_f)\bar{z})$ uses clause (1) on page 5. Recall that $\bar{S} = (q_0, z, q_f)$. $(\bar{q}_1, abb, (q_0, z, q_f)\bar{z}) \vdash (\bar{q}_1, bb, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_f)\bar{z})$ uses the first entry on the 3rd line of clause (7c) on page 7. $(\bar{q}_1, bb, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_f)\bar{z}) \vdash (\bar{q}_1, b, (q_1, 1, q_1)(q_1, z, q_f)\bar{z})$ uses clause (4) on page 6. $(\bar{q}_1, b, (q_1, 1, q_1)(q_1, z, q_f)\bar{z}) \vdash (\bar{q}_1, \lambda, (q_1, z, q_f)\bar{z})$ uses clause (5) on page 6. $(\bar{q}_1, \lambda, (q_1, z, q_f)\bar{z}) \vdash (\bar{q}_1, \lambda, \bar{z})$ uses clause (6) on page 6. $(\bar{q}_1, \lambda, \bar{z}) \vdash (\bar{q}_f, \lambda, \lambda)$ uses clause (2) on page 5.

Now let’s take a closer look at the second move:

$$(\bar{q}_1, abb, (q_0, z, q_f)\bar{z}) \vdash (\bar{q}_1, bb, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_f)\bar{z}).$$

Again, this uses the first entry on the 3rd line of clause (7c) on page 7. In other words, it uses clause template (7) on page 6 with $q_x = q_f$, $q_y = q_1$, and $q_z = q_1$. Why did I choose these values for q_x , q_y and q_z ?

Notice that before making the second move \bar{M} has (q_0, z, q_f) on the top of it’s stack. We are using \bar{M} to simulate M . The q_0 in (q_0, z, q_f) represents the fact that M is in state q_0 at the beginning of its run. The z in (q_0, z, q_f) represents the fact that M has a z on the top of its stack. The q_f in (q_0, z, q_f) represents the fact that when M pops the z off of its stack (later, at the end of its calculation) then M will be in state q_f .

Now the first move of M is: $(q_0, abb, z) \vdash (q_0, bb, 11z)$. We are trying to simulate this move with \bar{M} , so our next move in \bar{M} is to $(\bar{q}_1, bb, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_f)\bar{z})$. M reads an a from input, so \bar{M} reads an a from input. M moves to state q_0 . So \bar{M} now has q_0 as the left-most component of the upper entry on its stack, which is now $(q_0, 1, q_1)$. M pushed $11z$ onto its stack. This is represented by the fact that the middle components of the three items that \bar{M} pushed onto its stack is exactly $11z$.

It is now easy to see why I chose $q_x = q_f$. This was necessary because there was (q_0, z, q_f) on the top of the stack in \bar{M} and so I needed to choose $q_x = q_f$ in order to have clause (7) apply at all. But why did I choose the particular values for q_y and q_z ? Clause (7) would be applicable for any values of q_y and q_z , as long as $q_x = q_f$.

So why did I choose $q_x = q_y = q_1$? Well, keep in mind what I am trying to do. I am trying to give a sequence of instantaneous descriptions for \bar{M} which simulates M . So what I did is this: I looked down the line at what M was planning to do in the future. In particular, I asked myself the question: What state is M going to be in when its stack contents are again reduced back down to: $1z$? I saw that the answer is q_1 . I then asked myself: What state is M going to be in when its stack contents are again reduced back down to: z ? I saw that the answer again is q_1 . This explains why I chose $x = 1$ and $y = 1$.

To understand this further, let's now look at \bar{M} 's third move, which simulates M 's second move.

M 's move is: $(q_0, bb, 11z) \vdash (q_1, b, 1z)$. That is, M reads b , pops 1 , and switches to state q_1 . The corresponding move in \bar{M} is:

$$(\bar{q}_1, bb, (q_0, 1, q_1)(q_1, 1, q_1)(q_1, z, q_f)\bar{z}) \vdash (\bar{q}_1, b, (q_1, 1, q_1)(q_1, z, q_f)\bar{z}).$$

That is \bar{M} reads b , pops $(q_0, 1, q_1)$, and remains in state \bar{q}_1 . After performing this move, the top of the stack in \bar{M} contains $(q_1, 1, q_1)$. The left-most component of this is q_1 . This represents the fact that M is in state q_1 . But M really is in state q_1 . The reason this worked, that is, the reason that \bar{M} is still representing M even after the pop operation, is because I had the foresight in the previous move to choose $x = 1$. Analyzing the next moves of M and \bar{M} similarly explains why I chose $y = 1$.

In summary, \bar{M} is simulating M . Each move by M which increases the size of its stack, corresponds to a whole set of possible moves by \bar{M} . We want to see that \bar{M} is equivalent to M . Given a sequence of instantaneous descriptions of M which accepts a particular string w , we want to see that there is some corresponding sequence of instantaneous descriptions of \bar{M} which accepts w . So we are allowed to cleverly choose exactly the right sequence of instantaneous descriptions of \bar{M} . We do this by looking down the line at what M is going to do in the future. Every time we increase the size of the stack in \bar{M} , we have choices to make. We make those choices so that later on, when the size of the stack has been reduced back down to what it was before the move, \bar{M} is still successfully simulating M .