Non-Deterministic Push Down Automata

Let’s give our automata memory access! (For now, just a stack.)
Non-Deterministic Push Down Automata

Let's give our automata memory access! (For now, just a stack.)
Non-Deterministic Push Down Automata

Let’s give our automata memory access! (For now, just a stack.)
Non-Deterministic Push Down Automata

Let’s give our automata memory access! (For now, just a stack.)

\[ \Gamma \] is the stack alphabet. In this case, \( \Gamma = \{ \$, X \} \)
Non-Deterministic Push Down Automata

Let’s give our automata memory access! (For now, just a stack.)

$\Gamma$ is the stack alphabet. In this case, $\Gamma = \{\$, X\}$

For transition labeled $a$, $X/XX$:

- You can take that transition only if the next input character is $a$, AND the top of the stack holds $X$.
- You remove the $X$ and replace it with $XX$. 
Non-Deterministic Push Down Automata

Let’s give our automata memory access! (For now, just a stack.)

Γ is the stack alphabet. In this case, \( \Gamma = \{\$, X\} \)

For transition labeled \( a, \Lambda/XX \):

- You can take that transition if the next input character is \( a \), *regardless* of what’s on the stack (it does not have to be empty).
- You remove nothing from the stack and push \( XX \).
Non-Deterministic Push Down Automata

Let’s give our automata memory access! (For now, just a stack.)

\[ \Gamma \] is the stack alphabet. In this case, \( \Gamma = \{\$, X\} \)

For transition labeled \( b, X/\Lambda \):

- You can take that transition if the next input character is \( b \), AND the top of the stack holds \( X \).
- You pop the \( X \) and push nothing.

\( \Gamma \) is the stack alphabet. In this case, \( \Gamma = \{\$, X\} \)

For transition labeled \( b, X/\Lambda \):
Non-Deterministic Push Down Automata

Let's give our automata memory access! (For now, just a stack.)

$\Gamma$ is the stack alphabet. In this case, $\Gamma = \{\$, X\}$

What is the language of this NPDA?
NPDA: Palindromes
NPDA: Palindromes

\[
\rightarrow \quad a, \lambda / A \\
\frac{b, \lambda / B}{a, \lambda / a} \\
b, \lambda / b
\]
NPDA: Palindromes
NPDA: Palindromes

\[
\begin{align*}
\rightarrow & \quad \overset{a, \varepsilon/A}{\overset{b, \varepsilon/B}{\overset{a, \varepsilon/a}{\overset{b, \varepsilon/b}{\overset{a, a/\varepsilon}{\overset{b, b/\varepsilon}{\text{accept}}}}}}}
\end{align*}
\]
NPDA: Palindromes

\[
\begin{align*}
\rightarrow & a, \lambda / A \\
\quad & b, \lambda / B \\
\rightarrow & a, a / \lambda \\
\quad & b, b / \lambda \\
\rightarrow & a, A / \lambda \\
\quad & b, B / \lambda
\end{align*}
\]
NPDA: Palindromes
Separating DPDAs and NPDAs

Theorem
There exists a context free language that is not recognizable by any deterministic push-down automata.

$L = \{ a^i b^i | i > 0 \} \cup \{ a^i b^{2i} | i > 0 \}$

Examples: $aabb, aabbbb, aaabbb, aaabbbbbb \in L$

Exercise: prove that this a CFL.
Separating DPDAs and NPDAs

Theorem

There exists a context free language that is not recognizable by any deterministic push-down automata.

\[ L = \{ a^i b^i \mid i > 0 \} \cup \{ a^i b^{2i} \mid i > 0 \} \]
Theorem

There exists a context free language that is not recognizable by any deterministic push-down automata.

\[ L = \{a^ib^i \mid i > 0\} \cup \{a^ib^{2i} \mid i > 0\} \]

Examples: \(aabb, aabbbb, aaabbb, aaabbbbb \in L\)

Exercise: prove that this a CFL.

Suppose some DPDA, \(M\), recognizes \(L\).

We will show how to use \(M\) to construct DPDA \(M'\) that recognizes \(\{a^ib^ic^i \mid i > 0\}\).

Since we know that this language is not a CFL, we've arrived at a contradiction.

Therefore, \(M\) must not exist.
Separating DPDAs and NPDAs

Theorem

There exists a context free language that is not recognizable by any deterministic push-down automata.

\[ L = \{ a^i b^i \mid i > 0 \} \cup \{ a^i b^{2i} \mid i > 0 \} \]

Examples: \( aabb, aabbbb, aaabbb, aaabbbbb \in L \)

Exercise: prove that this a CFL.
Separating DPDAs and NPDAs

Theorem

There exists a context free language that is not recognizable by any deterministic push-down automata.

\[ L = \{a^i b^i | i > 0\} \cup \{a^i b^{2i} | i > 0\}\]

Examples: \(aabb, aabbbb, aaabbb, aaabbbbb \in L\)

Exercise: prove that this a CFL.

Suppose some DPDA, \(M\), recognizes \(L\).
Separating DPDAs and NPDAs

Theorem

There exists a context free language that is not recognizable by any deterministic push-down automata.

$L = \{a^i b^i \mid i > 0\} \cup \{a^i b^{2i} \mid i > 0\}$ Examples: $aabb, aabbbb, aaabbb, aaabbbbbb \in L$

Exercise: prove that this a CFL.

Suppose some DPDA, $M$, recognizes $L$.

We will show how to use $M$ to construct DPDA $M'$ that recognizes $\{a^i b^i c^i \mid i > 0\}$. 
Separating DPDAs and NPDAs

Theorem

There exists a context free language that is not recognizable by any deterministic push-down automata.

$L = \{a^i b^i \mid i > 0\} \cup \{a^i b^{2i} \mid i > 0\}$

Examples: $aabb, aabbbb, aaabbb, aaabbbbbbb \in L$

Exercise: prove that this a CFL.

Suppose some DPDA, $M$, recognizes $L$.

We will show how to use $M$ to construct DPDA $M'$ that recognizes $\{a^i b^i c^i \mid i > 0\}$.

Since we know that this language is not a CFL, we’ve arrived at a contradiction.

Therefore, $M$ must not exist.
Separating DPDAs and NPDAs

**Theorem**

There exists a context free language that is not recognizable by any *deterministic* push-down automata.

$L = \{a^i b^i \mid i > 0\} \cup \{a^i b^{2i} \mid i > 0\}$

Examples: $aabb, aabbbb, aaabbb, aaabbbbb$ \( \in L \)

Exercise: prove that this a CFL.

Suppose some DPDA, $M$, recognizes $L$. Take 2 copies of $M$: $M_1, M_2$
Separating DPDAs and NPDAs

Theorem

There exists a context free language that is not recognizable by any deterministic push-down automata.

\[ L = \{a^i b^i \mid i > 0\} \cup \{a^i b^{2i} \mid i > 0\} \]

Examples: \(aabb, aabbbb, aaabbb, aaabbbb \in L\)

Exercise: prove that this a CFL.

Suppose some DPDA, \(M\), recognizes \(L\). Take 2 copies of \(M\): \(M_1, M_2\)

We construct \(M'\) by combining the 2 copies in a particular way.
We claim \(M'\) recognizes \(\{a^i b^i c^i \mid i > 0\}\)
Separating DPDAs and NPDAs

\[ L = \{a^i b^i \mid i > 0\} \cup \{a^i b^{2i} \mid i > 0\} \]

Examples: \(aabb, aabbbb, aaabbb, aaabbbbb\in L\)

Suppose some DPDA, \(M\), recognizes \(L\). Take 2 copies of \(M\): \(M_1, M_2\)

We construct \(M'\) by combining the 2 copies in a particular way.

We claim \(M'\) recognizes \(\{a^i b^i c^i \mid i > 0\}\)

1. The start state of \(M'\) is the start state of \(M_1\).
Separating DPDAs and NPDAs

$L = \{a^i b^i \mid i > 0\} \cup \{a^i b^{2i} \mid i > 0\}$  Examples:  $aabb, aabbbb, aaabbb, aaabbbbbbb \in L$

Suppose some DPDA, $M$, recognizes $L$. Take 2 copies of $M$: $M_1, M_2$

We construct $M'$ by combining the 2 copies in a particular way. We claim $M'$ recognizes $\{a^i b^i c^i \mid i > 0\}$

1. The start state of $M'$ is the start state of $M_1$.
2. The accepting states in $M'$ are the accepting states of $M_2$ (and not $M_1$).
Separating DPDAs and NPDAs

\[ L = \{a^i b^i \mid i > 0\} \cup \{a^i b^{2i} \mid i > 0\} \]

Examples: \(aabb, aabbbb, aaabbb, aaabbbbbb \in L\)

Suppose some DPDA, \(M\), recognizes \(L\). Take 2 copies of \(M\): \(M_1, M_2\)

We construct \(M'\) by combining the 2 copies in a particular way. We claim \(M'\) recognizes \(\{a^i b^i c^i \mid i > 0\}\)

1. The start state of \(M'\) is the start state of \(M_1\).
2. The accepting states in \(M'\) are the accepting states of \(M_2\) (and not \(M_1\)).
3. If \(M_1\) has a transition from some accepting state \(p\) to (any) state \(q\) on input character \(b\), regardless of the stack instruction, create an additional transition from \(p\) in \(M_1\) to \(q\) in \(M_2\) and label it \(c\).
Separating DPDAs and NPDAs

\[ L = \{ a^i b^i \mid i > 0 \} \cup \{ a^i b^{2i} \mid i > 0 \} \]

Examples: \( aabb, aabbbb, aaabbb, aaabbbbbbb \in L \)

Suppose some DPDA, \( M \), recognizes \( L \). Take 2 copies of \( M \): \( M_1, M_2 \)

We construct \( M' \) by combining the 2 copies in a particular way.
We claim \( M' \) recognizes \( \{ a^i b^i c^i \mid i > 0 \} \)

1. The start state of \( M' \) is the start state of \( M_1 \).
2. The accepting states in \( M' \) are the accepting states of \( M_2 \) (and not \( M_1 \)).
3. If \( M_1 \) has a transition from some accepting state \( p \) to (any) state \( q \) on input character \( b \), regardless of the stack instruction, create an additional transition from \( p \) in \( M_1 \) to \( q \) in \( M_2 \) and label it \( c \).
4. In \( M_2 \), replace all \( b \)'s to \( c \)'s.
Separating DPDAs and NPDAs
Separating DPDAs and NPDAs
Separating DPDAs and NPDAs
Separating DPDAs and NPDAs
Separating DPDAs and NPDA
Separating DPDA\$s$ and NPDA\$s$
Separating DPDAs and NPDAs