Regulated Nondeterminism in PDAs: The Non-Regular Case

Tomáš Masopust

Faculty of Information Technology
Brno University of Technology
Czech Republic

NCMA 2009
Wroclaw, Poland
Transitions used:
Transitions used: \( r_1 \)
Pushdown Automaton

Transitions used: $r_1, r_2$
Regulated PDAs (Kolář and Meduna, 2000)

- Motivated by regulations in grammars.
- Given a PDA $M$ and a control language $R$.

Transitions used: $r_1, r_2, \ldots, r_k$

- It accepts the input (by a final state) if $M$ accepts the input and $r_1 r_2 \ldots r_k \in R$. 
Regulated PDAs (Kolář and Meduna, 2000)

- Motivated by regulations in grammars.
- Given a PDA $M$ and a control language $R$.

Transitions used: $r_1, r_2, \ldots, r_k$

It accepts the input (by a final state) if $M$ accepts the input and $r_1r_2\ldots r_k \in R$. 
Regulated PDAs (Kolář and Meduna, 2000)

- Motivated by regulations in grammars.
- Given a PDA $M$ and a control language $R$.

Transitions used: $r_1, r_2, \ldots, r_k$

It accepts the input (by a final state) if $M$ accepts the input and $r_1 r_2 \ldots r_k \in R$. 

Regulated PDAs (Kolář and Meduna, 2000)

- Regulated PDAs with regular control languages are ordinary PDAs.

- Regulated PDAs with non-regular (linear) control languages are computationally complete.
Given a PDA $M$ and a regular control language $R$. It accepts the input if $M$ accepts and $b_1 \ldots b_k \in R$ (in each step). Equivalent to ordinary pushdown automata.
Regularly Regulated Pushdowns (Křivka, 2007)

- Given a PDA $M$ and a regular control language $R$.
- It accepts the input if $M$ accepts and $b_1 \ldots b_k \in R$ (in each step).
- Equivalent to ordinary pushdown automata.
Given a PDA $M$ and a regular control language $R$. It accepts the input if $M$ accepts and $b_1 \ldots b_k \in R$ (in each step).

Equivalent to ordinary pushdown automata.
Generalization: considering nondeterminism.

Given a PDA $M$ and a control language $R$,

Next step is

\[ b_1 \ldots b_k \in R \quad \text{nondeterministic} \]
\[ b_1 \ldots b_k \notin R \quad \text{deterministic} \]
Generalization: considering nondeterminism.

Given a PDA $M$ and a control language $R$

Next step is

\[
\begin{cases}
    b_1 \ldots b_k \in R & \text{nondeterministic} \\
    b_1 \ldots b_k \not\in R & \text{deterministic}
\end{cases}
\]
Definition

Given a PDA

\[ \mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F) \]

and a control language \( R \subseteq (\Gamma \setminus Z_0)^* \). \( \mathcal{M} \) is an \( R \)-PDA if:

1. for all \( q \in Q, a \in \Sigma \cup \{\lambda\}, \) and \( Z \in \Gamma, \) \( \delta \) can be written as

\[
\delta(q, a, Z) = \delta_d(q, a, Z) \cup \delta_{nd}(q, a, Z),
\]

where \( d \) = deterministic and \( nd \) = nondeterministic, and

2. for all \( q, q' \in Q, a \in \Sigma \cup \{\lambda\}, w \in \Sigma^*, \) \( Z \in \Gamma, \) and \( \gamma \in \Gamma^*, \)

\[(q, aw, Z \gamma) \vdash_{\mathcal{M}} (q', w, \gamma' \gamma)\]

if

1. either \((q', \gamma') \in \delta_{nd}(q, a, Z), Z \gamma = \gamma'' Z_0, \) and \((\gamma'')^R \in R,\)

2. or \( \delta_d(q, a, Z) = (q', \gamma'), Z \gamma = \gamma'' Z_0, \) and \((\gamma'')^R \notin R.\)
Definition

Given a PDA

\[ M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F) \]

and a control language \( R \subseteq (\Gamma \setminus Z_0)^* \). \( M \) is an \( R \)-PDA if:

1. for all \( q \in Q \), \( a \in \Sigma \cup \{\lambda\} \), and \( Z \in \Gamma \), \( \delta \) can be written as

\[
\delta(q, a, Z) = \delta_d(q, a, Z) \cup \delta_{nd}(q, a, Z),
\]

where \( d = \text{deterministic} \) and \( nd = \text{nondeterministic} \), and

2. for all \( q, q' \in Q \), \( a \in \Sigma \cup \{\lambda\} \), \( w \in \Sigma^* \), \( Z \in \Gamma \), and \( \gamma \in \Gamma^* \),

\[(q, aw, Z\gamma) \vdash_M (q', w, \gamma'\gamma) \text{ if}
\]

1. either \((q', \gamma') \in \delta_{nd}(q, a, Z)\), \( Z\gamma = \gamma''Z_0 \), and \((\gamma'')^R \in R\),

2. or \( \delta_d(q, a, Z) = (q', \gamma') \), \( Z\gamma = \gamma''Z_0 \), and \((\gamma'')^R \notin R\).
Definition

Given a PDA

\[ M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F) \]

and a control language \( R \subseteq (\Gamma \setminus Z_0)^* \). \( M \) is an \( R \)-PDA if:

1. for all \( q \in Q, a \in \Sigma \cup \{\lambda\}, \text{ and } Z \in \Gamma, \delta \) can be written as

\[ \delta(q, a, Z) = \delta_d(q, a, Z) \cup \delta_{nd}(q, a, Z), \]

where \( d = \) deterministic and \( nd = \) nondeterministic, and

2. for all \( q, q' \in Q, a \in \Sigma \cup \{\lambda\}, w \in \Sigma^*, Z \in \Gamma, \text{ and } \gamma \in \Gamma^*, \)

\[ (q, aw, Z\gamma) \vdash M (q', w, \gamma'\gamma) \]

if

\[ \text{either } (q', \gamma') \in \delta_{nd}(q, a, Z), Z\gamma = \gamma''Z_0, \text{ and } (\gamma'')^R \in R, \]

\[ \text{or } \delta_d(q, a, Z) = (q', \gamma'), Z\gamma = \gamma''Z_0, \text{ and } (\gamma'')^R \notin R. \]
Example – $R = \{a^n b^n : n \geq 1\}$

$M$ nondeterministically checks that $Z_0 a^m b^n \in R$, i.e., $m = n$. 

$T(M) = \{a^n b^n : n \geq 1\}$. 

State: $q_a$
Example – \( R = \{ a^n b^n : n \geq 1 \} \)

\[ \begin{array}{cccccccc}
& & & & & & & \text{tape} \\
& & a & a & b & b & c & c & d & d \\
\end{array} \]

State: \( q_a \)

\( M \) nondeterministically checks that \( Z_0 a^m b^n \in R \), i.e., \( m = n \).
Example – $R = \{a^n b^n : n \geq 1\}$

$M$ nondeterministically checks that $Z_0 a^m b^n \in R$, i.e., $m = n$. 

State: $q_a$
Example – $R = \{a^n b^n : n \geq 1\}$

- $M$ nondeterministically checks that $Z_0 a^m b^n \in R$, i.e., $m = n$. 
Example – \( R = \{ a^n b^n : n \geq 1 \} \)

\[ \text{tape} \]
\[ a \ a \ b \ b \ c \ c \ d \ d \]

\[ \text{pushdown} \]
\[ b \]
\[ b \]
\[ a \]
\[ a \]
\[ Z_0 \]

State: \( q_b \)

- \( M \) nondeterministically checks that \( Z_0 a^m b^n \in R \), i.e., \( m = n \).
Example – \( R = \{ a^n b^n : n \geq 1 \} \)

\[ M \] nondeterministically checks that \( Z_0 a^m b^n \in R \), i.e., \( m = n \).
Example – $R = \{a^n b^n : n \geq 1\}$

- $M$ nondeterministically checks that $Z_0 a^m b^n \in R$, i.e., $m = n$. 

State: $q_c$
Example – $R = \{a^n b^n : n \geq 1\}$

$\mathcal{M}$ nondeterministically checks that $Z_0 a^m b^n \in R$, i.e., $m = n$. 

State: $q_d$
Example – $R = \{ a^n b^n : n \geq 1 \}$

$M$ nondeterministically checks that $Z_0 a^m b^n \in R$, i.e., $m = n$. 

State: $q_d$
Example – \( R = \{a^n b^n : n \geq 1\} \)

\[ \begin{array}{cccccccc}
\text{tape} & a & a & b & b & c & c & d & d \\
\text{pushdown} & & & & & & & & \\
\end{array} \]

\( \mathcal{M} \) nondeterministically checks that \( Z_0 a^m b^n \in R \), i.e., \( m = n \).

\[ T(\mathcal{M}) = \{a^n b^n c^n d^n : n \geq 1\}. \]
Properties

- \( R \)-PDAs behave nondeterministically iff their pushdown content forms a string belonging to \( R \).
- If \( R \) is regular, then the power of PDAs increases.

**Theorem**

Let \( R \) be a regular language and \( M \) be an \( R \)-PDA. Then, an equivalent PDA \( M' \) can effectively be constructed.

- If \( R \) is linear, then the power increases.
- What is the power of \( R \)-PDAs with non-regular control languages?
Properties

- $R$-PDAs behave nondeterministically iff their pushdown content forms a string belonging to $R$.

- $R$ is regular, then the power of PDAs.

Theorem

Let $R$ be a regular language and $\mathcal{M}$ be an $R$-PDA. Then, an equivalent PDA $\mathcal{M}'$ can effectively be constructed.

- $R$ is linear, then the power increases.

- What is the power of $R$-PDAs with non-regular control languages?
Properties

- $R$-PDAs behave nondeterministically iff their pushdown content forms a string belonging to $R$.

- $R$ is regular, then the power of PDAs.

**Theorem**

Let $R$ be a regular language and $M$ be an $R$-PDA. Then, an equivalent PDA $M'$ can effectively be constructed.

- $R$ is linear, then the power increases.

What is the power of $R$-PDAs with non-regular control languages?
Properties

- $R$-PDAs behave nondeterministically iff their pushdown content forms a string belonging to $R$.

- $R$ is **regular**, then the power of PDAs.

**Theorem**

Let $R$ be a regular language and $\mathcal{M}$ be an $R$-PDA. Then, an equivalent PDA $\mathcal{M}'$ can effectively be constructed.

- $R$ is **linear**, then the power increases.

- What is the power of $R$-PDAs with non-regular control languages?
Theorem

Let $L \in \text{RE}$. Then, there is a linear language $R$ and an $R$-PDA $\mathcal{M}$ such that

$$L = T(\mathcal{M}).$$
Proof (sketch) – $L^R = h(L_1 \cap L_2)$, $L_1, L_2$ linear

$\mathcal{M}$ nondeterministically pushes symbols onto its pushdown.
Proof (sketch) \(- L^R = h(L_1 \cap L_2), L_1, L_2\) linear

- \(M\) nondeterministically pushes symbols onto its pushdown.
Proof (sketch) – $L^R = h(L_1 \cap L_2), L_1, L_2$ linear

$M$ nondeterministically pushes symbols onto its pushdown.

\[ a_1 \ a_2 \ a_3 \ldots \ a_i \ \ldots \ldots \ a_n \]

\[ \begin{array}{c}
\text{tape} \\
\begin{array}{c}
Z_0 \\
b_1 \\
b_2 \\
b_3
\end{array}
\end{array} \]
Proof (sketch) – \( L^R = h(L_1 \cap L_2), \ L_1, L_2 \text{ linear} \)

\[ a_1 \ a_2 \ a_3 \ \cdots \ a_i \ \cdots \ a_n \]

- \( \gamma_{1} \in L_1 \)?
Proof (sketch) – $L^R = h(L_1 \cap L_2)$, $L_1$, $L_2$ linear

- $\gamma_1 \in L_1$ – YES; $\gamma_2 \in L_2$?
Proof (sketch) – $L^R = h(L_1 \cap L_2)$, $L_1$, $L_2$ linear

- $\gamma_1 \in L_1 \Rightarrow \text{YES}$; $\gamma_2 \in L_2 \Rightarrow \text{YES}$; we have $\gamma \in L_1 \cap L_2$. 
Proof (sketch) – $L^R = h(L_1 \cap L_2)$, $L_1, L_2$ linear

- Remove $b_k$, read $h(b_k)^R$ from the input.
Corollary

Let $L \in RE$. Then, there is a linear (deterministic) context-free language $R$ and an $R$-PDA

\[ \mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F) \]

such that $L = T(\mathcal{M})$,

- $|Q| \leq 3$,
- $|\Gamma| \leq |\Sigma| + 7$. 
Let
\[ M = (Q, \Sigma, \Gamma, \delta, q_0, Q_c, Z_0, F) \]
be a PDA, where \( Q_c \subseteq Q \) is a set of checking states. \( R \subseteq (\Gamma \setminus Z_0)^* \).

\( M \) is called a state-controlled \( R \)-PDA (\( R \)-sPDA) if for all \( q, q' \in Q \), \( a \in \Sigma \cup \{\lambda\} \), \( w \in \Sigma^* \), \( Z \in \Gamma \), and \( \gamma \in \Gamma^* \),

\[ (q, aw, Z\gamma) \vdash_M (q', w, \gamma') \]

if \( (q', \gamma') \in \delta(q, a, Z) \) and

1. either \( q \in Q \setminus Q_c \),
2. or \( q \in Q_c \), \( Z\gamma = \gamma''Z_0 \), and \((\gamma'')^R \in R\).
Theorem

Let $R$ be a regular language and $M$ be an $R$-sPDA. Then, an equivalent PDA $M'$ can effectively be constructed.
Theorem

Let $R$ be a regular language and $\mathcal{M}$ be an $R$-sPDA. Then, an equivalent PDA $\mathcal{M}'$ can effectively be constructed.

Theorem

Let $L \in RE$. Then, there is a linear language $R$ and an $R$-sPDA $\mathcal{M}$ such that

$$L = T(\mathcal{M}).$$

In addition, $\mathcal{M}$ checks the pushdown content no more than twice during any computation.
**Corollary**

Let $L \in RE$. Then, there is a linear language $R$ and an $R$-sPDA

$$M = (Q, \Sigma, \Gamma, \delta, q_0, Q_c, Z_0, F)$$

which checks the pushdown content no more than twice during any computation, such that

- $|Q| \leq 4$,
- $|Q_c| = 1$,
- $|\Gamma| \leq |\Sigma| + 6$,

and $L = T(M)$. 
Open Problems

By the example, there is a (deterministic) $R$-sPDA $\mathcal{M}$, where

- $R = \{a^n b^n : n \geq 1\}$ is linear, deterministic context-free,
- $T(\mathcal{M}) = \{a^n b^n c^n d^n : n \geq 1\}$,
- only one check of the pushdown content.
Open Problems

By the example, there is a (deterministic) $R$-sPDA $\mathcal{M}$, where

- $R = \{a^n b^n : n \geq 1\}$ is linear, deterministic context-free,
- $T(\mathcal{M}) = \{a^n b^n c^n d^n : n \geq 1\}$,
- only one check of the pushdown content.

Open Problem

What is the power of $R$-sPDAs with one check of the pushdown content and $R$ linear?
By the example, there is a (deterministic) $R$-sPDA $\mathcal{M}$, where

- $R = \{a^n b^n : n \geq 1\}$ is linear, deterministic context-free,
- $T(\mathcal{M}) = \{a^n b^n c^n d^n : n \geq 1\}$,
- only one check of the pushdown content.

**Open Problem**

What is the power of $R$-sPDAs with one check of the pushdown content and $R$ linear?

**Open Problem**

What is the power of deterministic $R$-sPDAs with $R$ linear?
Open Problems (Deterministic $R$-sPDAs)

- Deterministic $R$-sPDAs ($R$-sDPDAs), $R$ linear.
- $DCF \subset R$-sDPDA $\subseteq REC$ $(CS, detCS)$.
- Is $CF \subset R$-sDPDA?
Open Problems (Deterministic $R$-sPDAs)

- Deterministic $R$-sPDAs ($R$-sDPDAs), $R$ linear.
- $DCF \subset R$-sDPDA $\subseteq REC$ \((CS, \text{detCS})\).
- Is $CF \subseteq R$-sDPDA?
Open Problems (Deterministic $R$-sPDAs)

- Deterministic $R$-sPDAs ($R$-sDPDAs), $R$ linear.
- $DCF \subset R$-sDPDA $\subseteq REC \ (CS, \ detCS)$.
- Is $CF \subseteq R$-sDPDA?
Open Problems \((R\text{-PDAs}, R\text{-sPDAs}, \text{etc.})\)

- **Closure properties:** fix \(R\) and two \(R\text{-PDAs}\) \((R\text{-sPDAs})\) \(M_1\) and \(M_2\). Are the languages
  - \(T(M_1) \cup T(M_2)\)
  - \(T(M_1) \cap T(M_2)\)
  - \(T(M_1) \cdot T(M_2)\)
  - \(T(M_1)^*\) etc.

accepted by an \(R\text{-PDA}\) \((R\text{-sPDA})\)?

- **Decidable/undecidable problems.**
Open Problems ($R$-PDAs, $R$-sPDAs, etc.)

- **Closure properties:** fix $R$ and two $R$-PDAs ($R$-sPDAs) $\mathcal{M}_1$ and $\mathcal{M}_2$. Are the languages
  - $T(\mathcal{M}_1) \cup T(\mathcal{M}_2)$
  - $T(\mathcal{M}_1) \cap T(\mathcal{M}_2)$
  - $T(\mathcal{M}_1) \cdot T(\mathcal{M}_2)$
  - $T(\mathcal{M}_1)^*$ etc.

  accepted by an $R$-PDA ($R$-sPDA)?

- **Decidable/undecidable problems.**
Thank You