PARALLEL COMMUNICATING FLIP PUSHDOWN AUTOMATA SYSTEMS COMMUNICATING BY STACKS

M. Ramakrishnan
Research Scholar
Anna University of Technology, Coimbatore
Coimbatore – 641 047

S. Balasubramanian
Research Scholar
Anna University of Technology, Coimbatore
Coimbatore – 641 047

ABSTRACT

In this paper, we introduced parallel communicating flip pushdown automata communicating by stacks. PCFPA (2) equals the family of recursively enumerable languages. Also we have proved that. RPCFPA (3) equals the family of recursively enumerable languages.

Key words: Pushdown automata, Flip pushdown automata, finite automata

1. INTRODUCTION

A pushdown automaton is one way finite automaton with a separate pushdown store, that is first in first – out storage structure, which is manipulating by pushing and popping. Probably, such machines are best known for capturing the family of context free languages, which was independently established by Chomsky (Cole 1971). Pushdown automata have been extended in various ways.

The extension of pushdown automata is recently introduced, is called flip pushdown automata (Border et al 1982). A is an ordinary pushdown with the additional ability to flip its pushdown push down during the computation. This allows the machine to push and pop at both ends of the pushdown, therefore, a flip-pushdown is a form of a
deque storage structure and it becomes an equally power to Turing machines. A dequeue automaton cannot simulate two pushdowns.

DEFINITION 1.1: PARALLEL COMMUNICATING FLIP PUSHDOWN AUTOMATA

Parallel communicating flip pushdown automata system is of degree \( n \) is a construct

\[
A = (V, \Gamma, A_1, A_2, \ldots, A_n, K)
\]

where \( V \) is the input alphabet, \( \Gamma \) is the alphabet of pushdown symbols, for each \( 1 \leq i \leq n \),

\[
A_i = (Q_i, V, \Gamma_i, f_i, A_i, Z_i, F_i)
\]

where \( Q_i \) is a finite set of states, \( \Gamma_i \) is a finite pushdown alphabet, \( f_i \) is a transition mapping from \( V \cup \{\epsilon\} \times \Gamma_i \) to finite subsets of \( Q_i \times \Gamma_i^* \) is called transition function, \( \Delta_i \) is a mapping from \( Q_i \) to \( 2^{Q_i} \), \( q_i \in Q_i \), \( Z_i \in \Gamma_i \), \( F_i \subseteq Q_i \) be the set of final states and \( K \subseteq \{K_1, K_2, \ldots, K_n\} \subseteq \Gamma \) is the set of query symbols. The Flip Pushdown automata \( A_1, A_2, \ldots, A_n \) are components of the system \( A \).

If there exists only one component, \( A_i \), \( 1 \leq i \leq n \), such that \( (r, \alpha) \in f_i (q, a, A) \) with \( \alpha \in \Gamma^* \), \( |\alpha|_K > 0 \), for some \( r, q \in Q_i \), \( a \in V \cup \{\epsilon\} \), \( A \in \Gamma \), then the system is said to be centralized and \( A_i \) is said to be the master of the system. For the sake of simplicity, whenever a system is centralized its master is the first component.

A parallel communicating flip pushdown automata system is schematically represented in Figure 1.1.

As one can see in Figure 1.1, all stacks are connected with each other, each stack can send its contents to any other stack or receive the contents of any other stack. In the centralized case all stacks send their contents to a distinguished stack. In the case of the non-returning strategy, every stack preserves its contents after sending a copy of it to another component while in the case of the returning strategy the stack returns to its initial contents after communication.
The configuration or instantaneous of a parallel communicating flip pushdown automata system is defined as a 3n-tuple $(s_1, x_1, \alpha_1, s_2, x_2, \alpha_2, \ldots, s_n, x_n, \alpha_n)$

Where for $1 \leq i \leq n$,

$s_i \in Q_i$ is the current state of the component $A_i$,

$x_i \in V^*$ is the remaining part of the input word which has not yet been read by $A_i$,

$\alpha_i \in \Gamma^*$ is the contents of the ith stack, the rightmost symbol being the topmost symbol on the pushdown store

If $a$ is in $V \cup \{ \epsilon \}$, $x_i \in V^*$, $\alpha_i, \beta_i \in \Gamma^*$ and $Z_i$ is in $\Gamma$,

Then $(s_i, a, x_i, \alpha_i Z_i) \not\rightarrow (p_i, x_i, \alpha_i \beta_i)$, if the pair $(p_i, \beta_i)$ is in $f_i(s_i, a, x_i, \alpha_i)$, for the ordinary pushdown transitions and $(s_i, a, x_i, \alpha_i Z_i) \not\rightarrow (p_i, x_i, Z_i \alpha_i^R)$, if $p_i$ is in $\Delta_i(s_i)$, for the flip pushdown transition or a pushdown reversal transitions. Whenever, there is a choice between an ordinary pushdown transition or a pushdown reversal of any $i$, then the pushdown automaton non-deterministically chooses the next move. We do not
want the flip pushdown automaton to move the bottom of pushdown symbol when the pushdown is flipped.

We define two variants of transition relations on the set of configurations of A in the following way:

1. \((s_1, x_1, B_1 \alpha_1, ..., s_n, x_n, B_n \alpha_n) \vdash (p_1, y_1, \beta_1, ..., p_n, y_n, \beta_n)\)

Where \(B_i \in \Gamma, \alpha_i, \beta_i \in \Gamma^*, 1 \leq i \leq n\), iff one of the following two conditions holds:

   (i) \(K \cap \{B_1, B_2, ..., B_n\} = \emptyset\) and

   \[x_i = a_i y_i, a_i \in V \cup \{\varepsilon\}, \{p_i, \beta'_i\} \in f_i(s_i, a_i, B_i).\]

   \[\beta_i = \beta'_i \alpha_i, 1 \leq i \leq n,\]

   (ii) for all \(1 \leq i \leq n\) such that \(B_i = K_{ji}\) and \(B_{ji} \in K, \beta_i = B_j \alpha_{ji} \alpha_i,\)

   for all other \(r, 1 \leq r \leq n\), \(\beta_r = B_r \alpha_r,\) and

   \[y_i = x_i, p_t = s_t, \text{ for all } t, 1 \leq t \leq n.\]

2. \((s_1, x_1, B_1 \alpha_1, ..., s_n, x_n, B_n \alpha_n) \vdash (p_1, y_1, \beta_1, ..., p_n, y_n, \beta_n)\)

Where \(B_i \in \Gamma, \alpha_i, \beta_i \in \Gamma^*, 1 \leq i \leq n\), iff one of the following two conditions holds:

   (i) \(K \cap \{B_1, B_2, ..., B_n\} = \emptyset\) and

   \[x_i = a_i y_i, a_i \in V \cup \{\varepsilon\}, \{p_i, \beta'_i\} \in f_i(s_i, a_i, B_i).\]

   \[\beta_i = \beta'_i \alpha_i, 1 \leq i \leq n,\]

   (ii) for all \(1 \leq i \leq n\) such that \(B_i = K_{ji}\) and \(B_{ji} \notin K,\)

   for all other \(r, 1 \leq r \leq n\), \(\beta_r = B_r \alpha_r,\) and

   \[y_i = x_i, p_t = s_t, \text{ for all } t, 1 \leq t \leq n.\]

The two transition relations defined above differ when the topmost symbols of some stacks are execute a communication step replacing each query symbol with the requested stack contents of the corresponding component. If the topmost symbol of the queried stack is also a query symbol, then first this query symbol must be replaced with the contents of the corresponding stack. The top of each communicated stack must be a non-query symbol before its contents can be sent to another component. If this condition cannot be fulfilled, (a circular query appeared), then the work of the automation system is blocked.

After communication, the stack contents of the sending components remain the same in the case of relation, whereas it becomes the initial pushdown memory symbol.
in the case of relation. A parallel communicating flip pushdown automata system whose computations are based on relation is said to be non-returning; if its computations are based on relation, it is said to be returning.

The language accepted by a parallel communicating flip automata system A as above is defined as

$$\text{Rec}(A) = \{ x \in V^* | (q_1, x, Z_1, \ldots, q_n, x, Z_n) \vdash^* (s_1, \varepsilon, \alpha_1, \ldots, s_n, \varepsilon, \alpha_1, \ldots, s_n, \varepsilon, \alpha_n), \text{ with exactly } n \text{ pushdown reversals, for any } \alpha_i \in \Gamma^* \text{ and any } s_i \in F_i, 1 \leq i \leq n \},$$

$$\text{Rec}_r(A) = \{ x \in V^* | (q_1, x, Z_1, \ldots, q_n, x, Z_n) \vdash_r^* (s_1, \varepsilon, \alpha_1, \ldots, s_n, \varepsilon, \alpha_1, \ldots, s_n, \varepsilon, \alpha_n), \text{ with exactly } n \text{ pushdown reversals, for any } \alpha_i \in \Gamma^* \text{ and any } s_i \in F_i, 1 \leq i \leq n \},$$

where $\vdash^*$ and $\vdash_r^*$, denote the reflexive and transitive closure of $\vdash$ and $\vdash_r$ respectively. In the following we use the notations

$rcpcpfa(n)$ - for returning centralized parallel communicating flip pushdown automata systems of degree n,

$rpcfpa(n)$ - for returning parallel communicating flip pushdown automata systems of degree n,

$rpcfpa(n)$ - for centralized parallel communicating flip pushdown automata systems of degree n,

$pcfpa(n)$ - for parallel communicating flip pushdown automata systems of degree n.

If $x(n)$ is a type of automata system, then $X(n)$ is the class of languages accepted by pushdown automata systems of type $x(n)$. For example, $RCPCFPA(n)$ is the class of languages accepted by automata of type $rcpcpfa(n)$ (returning centralized parallel communicating flip pushdown automata systems of degree n).
The following examples shows to help the better understanding of the notions defined above.

**EXAMPLE 1.1**

Let us consider the following \( rpfcpca \) (4) given by the transition mappings of its components.

\[
\begin{align*}
\text{f}_1(q_1, \varepsilon, Z_1) &= \{ (s_1, a) \}, \\
\text{f}_1(s_1, \varepsilon, Z_1) &= \{ (p_1, Z_1) \}, \\
\text{f}_1(p_1, \varepsilon, Z_1) &= \{ (h_1, Z_1) \}, \\
\text{f}_1(h_1, \varepsilon, Z_1) &= \{ (r_1, Z_1), (t_1, K_3) \}, \\
\text{f}_1(t_1, \varepsilon, a) &= \{ (s_1, a) \}, \\
\text{f}_3(q_3, \varepsilon, Z_3) &= \{ (s_3, K_1) \}, \\
\text{f}_3(s_3, \varepsilon, a) &= \{ (p_3, K_2 a) \}, \\
\text{f}_3(p_3, \varepsilon, a) &= \{ (h_3, a) \}, \\
\text{f}_3(h_3, \varepsilon, a) &= \{ (q_3, a) \}, \\
\text{f}_3(h_3, \varepsilon, Z_3) &= \{ (r_3, Z_3) \}, \\
\text{f}_3(r_3, a, Z_3) &= \{ (r_3, Z_3) \}, \\
\text{f}_2(q_2, \varepsilon, Z_2) &= \{ (s_2, K_1) \}, \\
\text{f}_2(s_2, \varepsilon, a) &= \{ (p_2, a) \}, \\
\text{f}_2(p_2, \varepsilon, Z_2) &= \{ (h_2, Z_2) \}, \\
\text{f}_2(p_2, \varepsilon, Z_2) &= \{ (h_2, Z_2) \}, \\
\text{f}_2(r_2, a, Z_2) &= \{ (r_2, Z_2) \}, \\
\text{f}_4(s_4, \varepsilon, Z_4) &= \{ (h_4, Z_2) \}, \\
\text{f}_4(q_4, \varepsilon, Z_4) &= \{ (s_4, Z_4) \}, \\
\text{f}_4(h_4, \varepsilon, Z_4) &= \{ (t_4, Z_4), (v_4, K_3 Z_4) \}, \\
\text{f}_4(t_4, \varepsilon, Z_4) &= \{ (u_4, Z_4) \}, \\
\text{f}_4(v_4, \varepsilon, a) &= \{ (s_4, Z_4) \}, \\
\text{f}_4(v_4, \varepsilon, a) &= \{ (p_4, a) \}, \\
\text{f}_4(p_4, a, a) &= \{ (p_4, \varepsilon) \}, \\
\text{f}_4(p_4, \varepsilon, Z_4) &= \{ (r_4, Z_4) \}
\end{align*}
\]
The final states are $F_1 = \{r_1\}$, $F_2 = \{r_2\}$, $F_3 = \{r_3\}$, $F_4 = \{r_4\}$.

**EXAMPLE1.2**

A more intricate way of computation can be observed in the following cpcfpa (2).

\[
\begin{align*}
&f_1(q_1, X, Z_1) = \{(q_1, Z_1)\} \\
&f_1(s_1, \varepsilon, Z_2) = \{(p_1, \varepsilon)\} \\
&f_1(p_1, X, X) = \{(p_2, \varepsilon)\} \\
&f_1(p_2, \varepsilon, X) = \{(p_2, \varepsilon)\} \\
&f_1(p_2, \varepsilon, Z_2) = \{(p_1, \varepsilon)\} \\
&f_1(p_1, \varepsilon, Z_1) = \{(p_1, Z_1)\} \\
&f_2(q_2, X, Z_2) = \{(q_2, X, Z_2)\} \\
&f_1(q_1, c, Z_1) = \{(s_1, K_2 Z_1)\} \\
&f_2(s_2, X, Y) = \{(q_2, XY)\} \\
&f_1(s_1, \varepsilon, X) = \{(s_1, K_2 X)\} \\
&f_2(q_2, c, X) = \{(s_2, X)\} \\
&f_2(s_2, \varepsilon, X) = \{(s_2, \varepsilon)\} \\
&f_2(s_2, \varepsilon, Z_2) = \{(s_6, Z_2)\} \\
&f_2(s_6, \varepsilon, Z_2) = \{(s_6, Z_2)\} \\
&f_2(s_6, X, Z_2) = \{(s_6, Z_2)\}
\end{align*}
\]

where $X, Y \in \{a, b\}$, and the sets of final states are $F_1 = \{p_1\}$, $F_2 = \{s_1\}$.

**LEMMA1.1**

1. $\text{RCPCFPA}(n) \subseteq \text{RPCFPA}(n)$ and $\text{CPCFPA}(n) \subseteq \text{PCFPA}(n)$ for all $n \geq 1$.

2. $X(1)$ equals the family of context-free languages and $X(n) \subseteq X(n+1)$ for $X \in \{\text{RCPCFPA}, \text{RPCFPA}, \text{CPCFPA}, \text{PCFPA}\}$, $n \geq 1$.

**2 COMPUTATIONAL POWER**

We start by showing how non-returning parallel communicating pushdown automata systems can be simulated with returning systems.
THEOREM 2.1 \[ \text{PCFPA}(n) \subseteq \text{RPCFPA}(2n) \] for all \( n \geq 2 \).

PROOF

Let \( A = (a, \Delta, A_1, A_2, ..., A_n, K) \) be a pcpa\((n)\) with \( A_i = (Q_i, V, \Delta, f_i, q_i, Z_i, F_i) \), \( 1 \leq i \leq n \). We construct the rpcpa \((2n)\)

\[ A' = (V, \Gamma', A'_1, A'_2, ..., A'_n, K'), \]

Where \( K' = \{ K'_1, K'_2, ..., K'_n \} \cup \{ K_1, K_2, ..., K_n \} \), and

for all \( 1 \leq i \leq n \),

\[ A'_i = (Q_i \cup \{ q' | q \in Q_i \}, V, \Gamma \cup \{ \overline{K}_i \}, f'_i, q_i, Z_i, F_i), \]

with

\[ f'_i(q, a, A) = \{ (r', x) | (r, x) \in f_i(q, a, A) \} \]

(2) \[ f'_i(q', \epsilon, Z_i) = \{ (q, \overline{K}_i) \}, \]

where \( q, r \in Q_i, a \in V \cup \{ \epsilon \}, A \in \Gamma, \) and

\[ \overline{A}_i = (\{ \overline{q}_i \}, V, \Gamma \cup \{ \overline{K}_i \}, \overline{f}_i, \overline{q}_i, \overline{Z}_i \cup \{ \overline{q}_i \} \}, \]

with

(3) \[ \overline{f}_i(\overline{q}_i, a, \overline{Z}_i) = \{ (\overline{q}_i, K'_i) \}, \]

(4) \[ \overline{f}_i(\overline{q}_i, \epsilon, A) = \{ (\overline{q}_i, A) \}, \]

where \( a \in V \cup \{ \epsilon \}, A \in \Gamma \)

As one can easily see, every component of \( A \) has a “satellite” component in \( A' \). Each accepting step in \( A \) is simulated by two accepting and two communication steps in \( A' \) in the following way:

In the first accepting step \( A'_i \) and \( \overline{A}_i \) use the rules (1) and (3), respectively. Now the stacks of all components \( A'_i \) have the same contents as the corresponding ones in \( A \). Moreover, the current states of \( A'_i \) are copies of the current states of the corresponding ones in \( A \).
THEOREM 2.2

PCFPA (2) equals the family of recursively enumerable languages.

PROOF

Let $A = (Q, V, \Gamma, f, q, B_1, B_2, F)$ be a two-stack flip pushdown automaton.

We construct the $pcfpa(2)$

$$A = (V, \Gamma', A_1, A_2, \{K_1, K_2\})$$

with

$$\Gamma' = \Gamma \cup \{Z_i, [q, a, X, Y, \alpha, \beta]_i \mid q, \alpha, \beta \in f(r, a, X, Y), q, r, \in Q,$$

$$a \in V \cup \{\varepsilon\}, X, Y \in \Gamma, 1 \leq i \leq 2\},$$

and

$$A_i = \{Q_i, V, \Gamma', f_i, q_i, Z_i, K_i\}, 1 \leq i \leq 2$$

with

$$\Gamma' = \Gamma \cup \{Z_i, [q, a, X, Y, \alpha, \beta]_i \mid (q, \alpha, \beta) \in f(r, a, X, Y), q, r, \in Q,$$

$$a \in V \cup \{\varepsilon\}, X, Y \in \Gamma, 1 \leq i \leq 2\},$$

and

$$A_i = \{Q_i, V, \Gamma', f_i, q_i, Z_i, F'\}, 1 \leq i \leq 2$$

where

$$Q_i = \{q_{ch}, q_{rec}, q_w, [q, \alpha], [q, \beta] \mid (q, \alpha, \beta) \in f(r, a, X, Y), q, r, \in Q,$$

$$a \in V \cup \{\varepsilon\}, X, Y \in \Gamma\} \cup \{[q, X] \mid q \in Q, X \in \Gamma\} \cup \{q_1, q_2\},$$

and

$$F' = \{q_{ch}, q_{rec} \mid q \in F\}.$$

and the transition mappings are defined as follows:

$$1) \quad f_i(q_i, \varepsilon, Z_i) = \{(q_{ch} B_1 Z_1)\},$$

$$f_i(q_2, \varepsilon, Z_2) = \{(q_{rec} B_2 Z_2)\},$$

This is the initial step in which both components change their states and stack contents in order to begin the simulation.

$$2) \quad f_i(q_{ch} a, X) = \{([q_{ch} [r, a, X, Y, \alpha, \beta]]) \mid r, \alpha, \beta \in f(q, a, X, Y), Y \in \Gamma\},$$

$$f_i(q_{rec}, \varepsilon, X) = \{([q_{ch} K_i])\},$$

$$f_i(q_{rec}, \varepsilon, X) = \{([q_{ch} K_2])\},$$

$$f_i(q_{ch} a, Y) = \{([q_{ch} [r, a, X, Y, \alpha, \beta]]_2 \mid r, \alpha, \beta \in f(q, a, X, Y), X \in \Delta\},$$

where $a \in V \cup \{\varepsilon\}, X \in \Gamma$. 

42
The first automaton, being in a state $q_{\text{ch}}$, $q \in Q$, reading $a \in V \cup \{\varepsilon\}$ on its input tape and $X$ from its memory, chooses a possible move of the two-stack flip pushdown automaton in the same state, reading the same input symbol and reading $X$ from its first stack. This move is encoded in a flip pushdown memory symbol and stored in the memory of the first component. The other component, being in state $q_{\text{rec}}$, $q \in Q$ requests the memory contents of the first component without moving its reading head.

With the second group of transitions of (2) the same process can be done, but with $A_2$, choosing a possible transition of $A$ to be simulated. In simulation of $A_1$ and $A_2$ take turns in choosing the transitions.

\begin{align*}
(3) \quad f_1(q_{\text{ch}}, \varepsilon, [r, \alpha, X, Y, \alpha, \beta])_1 &= \{(r_w, \alpha)\}, \\
f_1(q_w, \varepsilon, X) &= \{(q_w, X), (q_{\text{rec}}, X)\}, \\
f_2(q, Y, \alpha, [r, \alpha, X, Y, \alpha, \beta])_1 &= \{([r], \beta, \varepsilon)\}, \\
f_2([q, \alpha], \varepsilon, X) &= \{(q, \alpha, \varepsilon)\}, \\
f_2([q, \alpha], \varepsilon, Z_1) &= \{(q_{\text{ch}}, \alpha)\}, \\
f_2(q_{\text{ch}}, \varepsilon, [r, \alpha, X, Y, \alpha, \beta])_2 &= \{([r], \beta)\}, \\
f_2(q_w, \varepsilon, X) &= \{(q_w, X), (q_{\text{rec}}, X)\}, \\
f_1(([q], X), \alpha, [r, \alpha, X, Y, \alpha, \beta])_2 &= \{([r], \beta, \varepsilon)\}, \\
f_1([q], \alpha, \varepsilon, X) &= \{(q, \alpha, \varepsilon)\}, \\
f_1([q], \alpha, \varepsilon, Z_2) &= \{(q_{\text{ch}}, \alpha)\}
\end{align*}

Where $q \in Q$ and $\alpha \in V \cup \{\varepsilon\}$.

as the non-returning variants.

**THEOREM 2.3**

$RPCFPA$ (3) equals the family of recursively enumerable languages.

**PROOF**

We first show that for every two-stack flip pushdown automaton.

Construct an $rpcfpa$ system with three components such that the two devices accept the same language.

Let $A = (Q, V, \Gamma, f, q_0, Z_0, Z_1, Z_2, F)$ be a two-stack flip pushdown automaton.
Let us associate to any transition \((r, \alpha, \beta) \in f(q, a, A, B)\), where \(q, r \in Q, a \in V \cup \{\varepsilon\}, A, B \in \Gamma, \alpha, \beta \in \Gamma^*\) a new symbol \([q, a, A, B, r, \alpha, \beta]\) and let us denote the set of these symbols by \(\Delta_t\).

Moreover, let \(Q' = \{q' \mid q \in Q\}\) and \(Q_c = \{\langle q, x, X \rangle \mid q \in Q, x \in V \cup \{\varepsilon\}, X \in \Gamma\}\).

The simulating \(rpcfpa\) \(A\) is constructed as follows:

\[ A = (V, \Gamma, A, A_1, A_2, A_3, \{K_1\}), \]

Where

\[ A_1 = (Q \cup Q', V, \Gamma_A, f_1, q_0, Z_1, F) \]
\[ A_2 = (Q \cup Q_c, V, \Gamma_A, f_2, q_0, Z_0^1, F) \]
\[ A_3 = (Q \cup Q_c, V, \Gamma_A, f_3, q_0, Z_0^2, F) \]

We define the transition mappings of the components as follows:

1. \(f_1(q, a, Z_1) = \{(r', [q, a, A, B, r, \alpha, \beta]) \mid [q, a, A, B, r, \alpha, \beta] \in \Gamma_t\}\),
\[ f_2(q, a, A) = \{\langle q, a, A \rangle, K_1\}\],
\[ f_3(q, a, B) = \{\langle q, a, B \rangle, K_1\}\],

where \(q \in Q, a \in V \cup \{\varepsilon\}, A, B \in \Gamma\),

2. \(f_1(r', \varepsilon, Z_1) = (r, Z_1)\),
\[ f_2((q, a, A), \varepsilon, [q, a, A, B, r, \alpha, \beta]) = (r, \alpha)\],
\[ f_3((q, a, B), \varepsilon, [q, a, A, B, r, \alpha, \beta]) = (r, \beta)\],

where \(r' \in Q', [q, a, A, B, r, \alpha, \beta] \in \Gamma_t\).

3. CONCLUSION

We define a property called communication parallel communicating flip pushdown automata. Flip pushdown automata are pushdown with the additional ability to flip or reverse its pushdown. We have proved that \(RPCFPA\) equals the family of recursively enumerable languages.
REFERENCES:


