Characterizations of Bounded Semilinear Languages by One-way and Two-way Deterministic Machines

Oscar H. Ibarra\textsuperscript{1} and Shinnosuke Seki\textsuperscript{2}

\textsuperscript{1}. Department of Computer Science, University of California, Santa Barbara, USA
\textsuperscript{2}. Department of Systems Biosciences for Drug Discovery, Kyoto University, Japan
Conceptual diagram of a counter machine

finite state control

input tape

a b

can be tested for zero

c_1 c_2 ... c_k

counters
Conceptual diagram of a counter machine

\[ \delta(p, a, 1, 1, \ldots, 0) = (q, R, 0, -1, \ldots, +1) \]
Variants of counter machine

- Pushdown counter machines
- 2-way counter machines
Formal definition of pushdown counter machines

For $k \geq 0$, a pushdown $k$-counter machine is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where

- $Q$ is a finite set of states
- $\Sigma, \Gamma$ are input and stack alphabets
- $Z_0$ is the bottom stack symbol
- $q_0$ is the initial state
- $F$ is a set of accepting states

and $\delta$ is a relation $Q \times \Sigma \times \Gamma \times \{0,1\}^k \to Q \times \{S,R\} \times \Gamma^* \times \{-1,0,+1\}^k$. 
Reversal-bounded counter machines

A finite automaton augmented with 2 counters is Turing-complete. [Minsky 1961]

Definition [Baker and Book 1974, Ibarra 1978]

For a positive integer $k \geq 1$, a counter machine is \textit{r-reversal} if for each of its counters, the number of alternations between non-decreasing mode and non-increasing mode – called \textit{reversal} – is upper-bounded by $r$ in any accepting computation.

* This equivalence needs a help of finite state control to remember how many reversals each counter has made.
**$r$-reversal counter and $1$-reversal counter**

One can simulate a counter that makes $r$-reversals by counters that make $1$ reversal.

**Lemma**

*For any positive integer $r \geq 1$, an $r$-reversal counter can be simulated by $\left\lfloor \frac{r+1}{2} \right\rfloor$ $1$-reversal counters.*

This lemma enables us to focus on the $1$-reversal counter machines.
Classes of counter machines

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCM($k$)</td>
<td>The class of deterministic $k$-counters machines</td>
</tr>
<tr>
<td>NCM($k$)</td>
<td>The class of (non-deterministic) $k$-counters machines</td>
</tr>
<tr>
<td>DPCM($k$)</td>
<td>The class of deterministic pushdown $k$-counters machines</td>
</tr>
<tr>
<td>NPCM($k$)</td>
<td>The class of pushdown $k$-counters machines</td>
</tr>
<tr>
<td>2DCM($k$)</td>
<td>The class of 2-way deterministic $k$-counters machines</td>
</tr>
</tbody>
</table>

For a machine class $M$, by $\mathcal{L}(M)$, we denote the set of all languages accepted by a machine in $M$. 
Classes of counter machines (cont.)

- $\text{DFA} = \text{DCM}(0)$
- $\text{NFA} = \text{NCM}(0)$
- $\text{DPDA} = \text{DPCM}(0)$
- $\text{NPDA} = \text{NPCM}(0)$
- $\text{DCM} = \bigcup_{k \geq 0} \text{DCM}(k)$ is the class of deterministic counter machines.

Note that $\mathcal{L}(\text{DFA}) = \mathcal{L}(\text{NFA}) \subsetneq \mathcal{L}(\text{DPDA}) \subsetneq \mathcal{L}(\text{NPDA})$. 
Semilinear sets

A subset $Q \subseteq N^k$ is a **linear set** if there exist $k$-dimensional integer vectors $v_0, v_1, \ldots, v_n$ such that

$$Q = \{v_0 + t_1 v_1 + \cdots + t_n v_n \mid t_1, \ldots, t_n \in N\}.$$ 

A set is **semilinear** if it is a finite union of linear sets.
Bounded semilinear languages

A language $L \subseteq \Sigma^*$ is **bounded** if there exist $k \geq 1$ and $x_1, x_2, \ldots, x_k \in \Sigma^+$ such that

$$L \subseteq x_1^* x_2^* \cdots x_k^*.$$ 

If $x_1, x_2, \ldots, x_k$ are letters, $L$ is especially said to be **letter-bounded**.

**Definition**

For a bounded language $L \subseteq x_1^* x_2^* \cdots x_k^*$, let

$$Q(L) = \{(i_1, \ldots, i_k) \mid x_1^{i_1} x_2^{i_2} \cdots x_k^{i_k} \in L\}.$$ 

If this set is semilinear, then we say that $L$ is **bounded semilinear**.
Finite-turn machines

Finite-turn

For an integer $t \geq 0$, a 2-way machine is $t$-turn if the machine can accept an input with at most $t$ “turns” of the input head. A 2-way machine is finite-turn if it is a $t$-turn for some $t \geq 0$.

Finite-crossing

For an integer $c \geq 1$, a 2-way machine is $c$-crossing if every accepted input admits a computation by the machine during which the input head crosses the boundary between any two adjacent symbols no more than $c$ times. A 2-way machine is finite-crossing if it is a $c$-turn for some $c \geq 1$. 
Main theorem

Main Theorem

The following 5 statements are equivalent for every bounded language $L$:

1. $L$ is bounded semilinear;
2. $L$ can be accepted by a finite-crossing 2DPCM;
3. $L$ can be accepted by a finite-turn 2DPDA whose stack is reversal-bounded;
4. $L$ can be accepted by a finite-turn 2DCM(1);
5. $L$ can be accepted by a 1DCM.
The following 5 statements are equivalent for every bounded language $L$:

1. $L$ is bounded semilinear;
2. $L$ can be accepted by a finite-crossing 2DPCM;
3. $L$ can be accepted by a finite-turn 2DPDA whose stack is reversal-bounded;
4. $L$ can be accepted by a finite-turn 2DCM(1);
5. $L$ can be accepted by a 1DCM.

$4 \rightarrow 3 \rightarrow 2$ and $5 \rightarrow 2$ trivially hold since

- finite-turn 2DCM(1) $\subseteq$ finite-turn 2DPDA $\subseteq$ finite-crossing 2DPCM, and
- 1DCM $\subseteq$ finite-crossing 2DPCM.

Let us prove

- $1 \rightarrow 4, 1 \rightarrow 5$
- $2 \rightarrow 1$
Proof direction for $1 \rightarrow 4, 1 \rightarrow 5$

<table>
<thead>
<tr>
<th>Accepted by</th>
<th>Language Accepted by</th>
<th>$L \subseteq a_1^* \cdots a_k^*$ (Q(L)) is semilinear</th>
<th>$L \subseteq x_1^* \cdots x_k^*$ (Q(L)) is semilinear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1NCM</td>
<td></td>
<td><strong>result 1</strong> [Ibarra 78]</td>
<td><strong>Use result 1 and non-deterministic guess.</strong></td>
</tr>
<tr>
<td>finite-turn 2DCM(1)</td>
<td></td>
<td>Linear Diophantine-equations turn out to be solvable by these machines</td>
<td>$1 \rightarrow 4, 1 \rightarrow 5$</td>
</tr>
<tr>
<td>or 1DCM</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Let $L \subseteq x_1^* \cdots x_k^*$ s.t. $Q(L)$ is semilinear.

Given a word $w$,

$$w \in L \iff \exists (i_1, \ldots, i_k) \in Q(L), w = x_1^{i_1} \cdots x_k^{i_k}.$$  

Let $L' = \{a_1^{i_1} \cdots a_k^{i_k} \mid (i_1, \ldots, i_k) \in Q(L)\}$, and construct a 1NCM $M'$ for it.

1NCM $M$ for $L$

1. reads $w$;
2. non-deterministically decomposes it as $w = x_1^{i_1} \cdots x_k^{i_k}$;
3. runs $M'$ on $a_1^{i_1} \cdots a_k^{i_k}$, and returns $M'$'s decision as its decision for $w$ and $L$.  

## Proof direction for $1 \rightarrow 4, 1 \rightarrow 5$

| Language Accepted by | $L \subseteq a_1^* \cdots a_k^*$  
$Q(L)$ is semilinear | $L \subseteq x_1^* \cdots x_k^*$  
$Q(L)$ is semilinear |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1NCM</td>
<td>result 1</td>
<td>Use result 1 and non-deterministic guess.</td>
</tr>
</tbody>
</table>
| finite-turn 2DCM(1)  
or 1DCM               | Linear Diophantine-equations turn out to be solvable by these machines | 1 \rightarrow 4, 1 \rightarrow 5 |

- How can a deterministic machine non-deterministic decompose an input?
- Our answer is “this decomposition does not require non-determinism.”
Deterministic decomposition

- Let $c$ be a sufficiently large constant.
  - This is efficiently computable from a given language $L \subseteq x_1^* \cdots x_k^*$ s.t. $Q(L)$ is semilinear.
- Let $L' = L \cap x_1 \geq c x_2 \geq c \cdots x_k \geq c$.

**Theorem [Fine and Wilf 1965]**

Let $u, v$ be primitive words. If $u^*$ and $v^*$ share their prefix of length $|u| + |v| - \gcd(|u|, |v|)$, then $u = v$.

- Based on this theorem, we can figure out that looking-ahead by some constant distance on an input tape settles the decomposition.
Main theorem

The following 5 statements are equivalent for every bounded language $L$:

1. $L$ is bounded semilinear;

2. $L$ can be accepted by a finite-crossing 2DPCM;

3. $L$ can be accepted by a finite-turn 2DPDA whose stack is reversal-bounded;

4. $L$ can be accepted by a finite-turn 2DCM(1);

5. $L$ can be accepted by a 1DCM.
Proof for $2 \rightarrow 1$

**Lemma**

*If a letter-bounded language is accepted by a finite-crossing 2DPDA $M$, then it is accepted by a finite-crossing 2DCM.*

**Proof idea**

- In a computation by a finite-crossing 2DPDA, contents of $M$'s stack consist of constant # of blocks of the form $uv^iw$ (pumped structure).
- Each of these blocks correspond to a writing phase during which the stack is never popped.
- # of “long” writing phases is bounded by a constant $c$.
- We build a 2DCM($c$), and let its counters to store $i_1, i_2, i_3, ...$, and let its finite state control to remember $(u_1, v_1, w_1), ...$
Proof for $2 \rightarrow 1$

The previous result is generalized for a broader class of finite-crossing 2DPCM.

<table>
<thead>
<tr>
<th>Lemma</th>
</tr>
</thead>
<tbody>
<tr>
<td>If a letter-bounded language is accepted by a finite-crossing 2DPCM, then it is accepted by a finite-crossing 2DCM.</td>
</tr>
</tbody>
</table>
Proof for 2 → 1

**Theorem**

A bounded language that is accepted by a finite-crossing 2DPCM is effectively semilinear.

**Proof.**

Let \( M \) be a finite-crossing 2DPCM(c) for some \( c \geq 0 \) s.t. \( L(M) \subseteq x_1^* \cdots x_k^* \).

1. It is easy to construct a finite-crossing 2DPCM(c) for \( \{a_1^{i_1} \cdots a_k^{i_k} \mid x_1^{i_1} \cdots x_k^{i_k} \in L(M)\} \).
2. This language is letter-bounded, and hence, this machine can be converted into an equivalent finite-crossing 2DCM.
3. It is known that if a letter-bounded language is accepted by a finite-crossing 2NCM, then the language is bounded semilinear.
4. Thus, \( Q(L(M)) \) is semilinear.
Main theorem

**Main Theorem (Proved!!)**

The following 5 statements are equivalent for every bounded language $L$:

1. $L$ is bounded semilinear;
2. $L$ can be accepted by a finite-crossing 2DPCM;
3. $L$ can be accepted by a finite-turn 2DPDA whose stack is reversal-bounded;
4. $L$ can be accepted by a finite-turn 2DCM(1);
5. $L$ can be accepted by a 1DCM.

**Open**

Is every bounded language accepted by a finite-crossing 2NPCM bounded semilinear.
Acknowledgements

This research was supported by

- National Science Foundation Grant
- Natural Sciences and Engineering Research Council of Canada Discovery Grant and Canada Research Chair Award in Biocomputing to Lila Kari
- Funding Program for Next Generation World-Leading Researchers (NEXT Program) to Yasushi Okuno
References

[Baker and Book 1974]

[Fine and Wilf 1965]

[Ibarra 1978]
O. H. Ibarra. Reversal-bounded multicounter machines and their decision problems. 
References

[Minsky 1961]

M. L. Minsky. Recursive unsolvability of Post’s problem of “tag” and other topics in theory of turing machines.