Recursive po2DFA: Hierarchical Automata for FO-definable Languages

Simoni S. Shah

Joint work with Paritosh K. Pandya
Tata Institute of Fundamental Research, Mumbai

9 Feb, 2015

Overview

- Recursive *po2dfa* and its properties
- Interesting Example Languages
  - The STAIR langues
  - The Bounded-Buffer languages
- The recursion hierarchy and FO equivalence
- A Temporal Logic for the recursion hierarchy
- Comparison with other FO hierarchies
- Summary and Interesting Questions

po2dfa

- Partially ordered - Only loops in transition graph are self-loops
- Single *Initial* (s), *Accept* (t), *Reject* (r) state
- The automaton loops in a given state, until a transition is enabled.
  Never comes back to that state.
- Two-way - On a transition, the head moves in either direction
- States are partitioned into left-moving and right-moving states.
  On a transition, head moves in the direction determined by the target state.
- Deterministic - Unique run on any given word
- The word is extended with end-markers: $\triangleright\ w\ \triangleleft$
- Notion of acceptance: $w, i \models \mathcal{M}$
  Pointed language of an automaton: $\{(w, i) \mid w, i \models \mathcal{M}\}$
  Language of and automataon: $\mathcal{L}(\mathcal{M}) = \{w \mid w, 1 \models \mathcal{M}\}$

Example

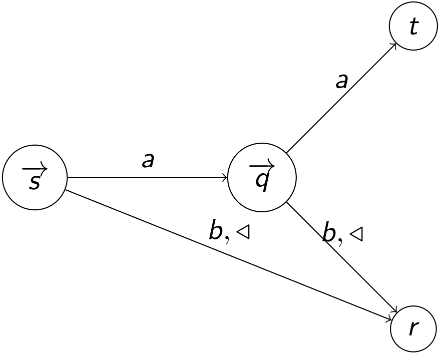


Figure: *po2dfa* $\mathcal{M}_{aa}$

This *po2dfa* accepts words which begin with two successive a's.

Recursive *po2dfa* or *Rpo2dfa*

- *Rpo2dfa*$[1]\ =\ $ *po2dfa*
- *Rpo2dfa*$[k]$ of *recursion depth* k
  - Partially ordered, Two-way, Deterministic
  - Transitions are guarded by Boolean functions of *Rpo2dfa*$[m]$, such that $m \leq k - 1$ (recursive)
    If $\mathscr{F} = \mathscr{B}(\mathcal{M}_j)$ is a boolean function of *Rpo2dfa* $\mathcal{M}_j$, assign $\top$ to $\mathcal{M}_j$ iff $w, i \models \mathcal{M}_j$

$$\overrightarrow{q_1} \xrightarrow{\qquad\mathscr{F}\qquad} \overrightarrow{q_2}$$

- For Determinism: Two transitions from the same state must have disjoint pointed languages
  $\forall w, i\ .\ w, i \not\models\ (\mathscr{F}_1 \wedge \mathscr{F}_2)$

Example

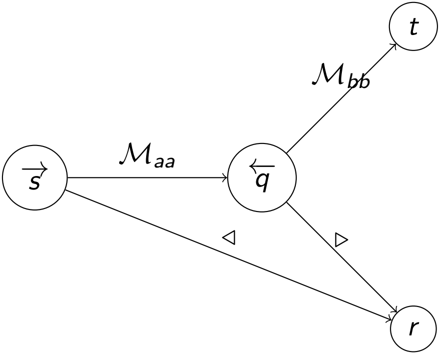


Figure: *Rpo2dfa*

w : $b\ a\ b\ a\ b\ b\ a\ b\ a\ a\ b\ b$

This *Rpo2dfa* accepts words which have a *bb* factor before its first *aa* factor.

The STAIR languages

- Consider the alphabet $\Sigma = \{a, b, c\}$
- $STAIR[k] \; = \; \Sigma^* \; (ac^*)^k a \; \Sigma^*$
  $k + 1$ occurrences of a without any b's between them.
- $STAIR[k] \in US^k$ and $STAIR[k] \notin US^{k-1}$

All STAIR[k] languages may be expressed using *Rpo2dfa*[2]

The STAIR languages

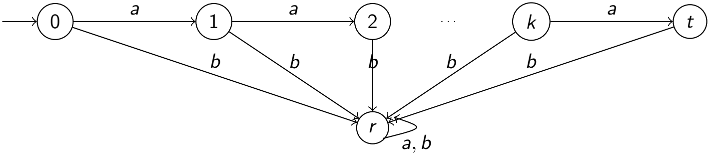
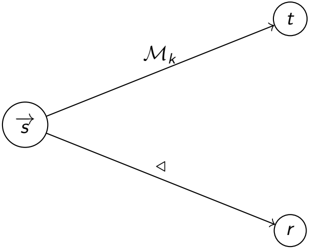


Figure: Automaton $\mathcal{M}_k$



The Bounded-Buffer Languages

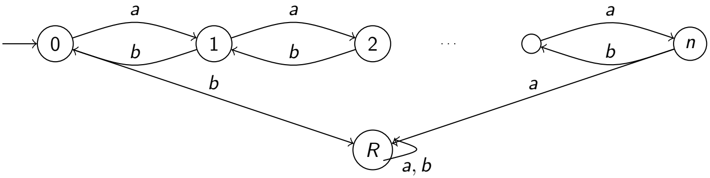


Figure: Bounded Buffer DFA of buffer size n - denoted BB_n

The Bounded-Buffer Languages

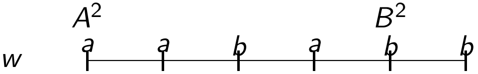
Consider any word $w \in \{a, b\}^*$. The BB_n accepts w if and only if

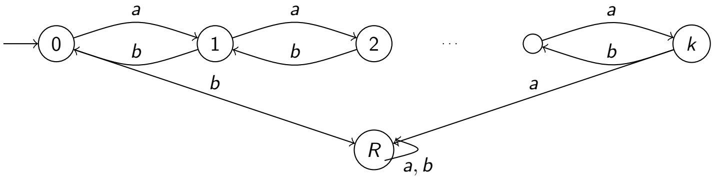
- No. of excessive a's must never exceed the limit n.
  i.e. $\#a(u) - \#b(u) \leq n$ for any prefix u.
- b's must never overtake a's.
  i.e. $\#b(u) \leq \#a(u)$ for any prefix u.
- $\#a(w) = \#b(w)$

Structure of a word over $\{a, b\}$

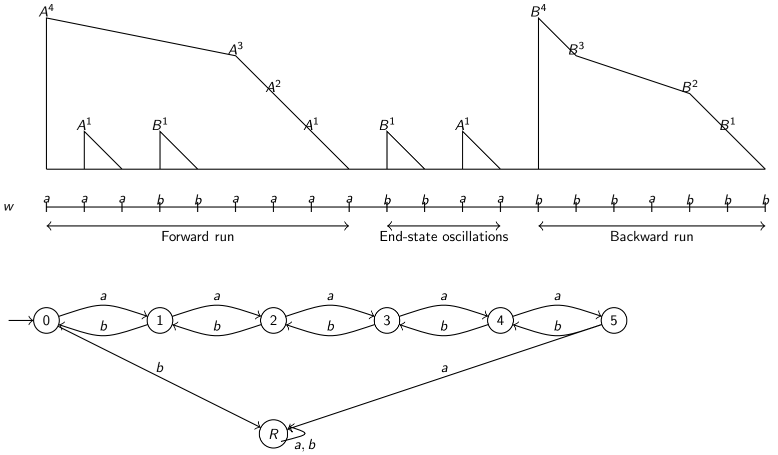
Mark each position in the word with its *scope index*:

- a scope index: Starting from 0, how far the DFA can go from that position, before returning to state 0.
- b scope index: What is the maximal state the run of the DFA can begin from, so that it reaches state 0, without reaching back to that state.





Structure of a word over $\{a, b\}$



Bounded Buffer Automaton

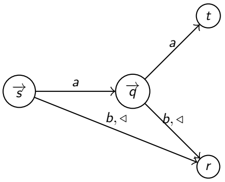


Figure: *po2dfa* $\mathcal{M}_{aa}$

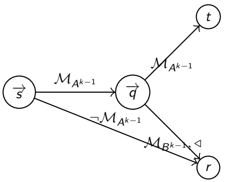


Figure: Automaton $\mathcal{M}_{A_k}$

The Bounded-Buffer Languages

Consider any word $w \in \{a, b\}^*$.

Theorem [PS15]

The BB_n accepts w if and only if

- No. of excessive a's must never exceed the limit n.
  $\nexists i \in dom(w).\ SI(w, i) = A_{n+1}$
- b's must never overtake a's.
  $\nexists i \in dom(w).\ SI(w, i) = B_{l+1} \wedge \forall j < i.\ SI(w, j) \leq A_l$
- $\#a(w) = \#b(w)$
  $\nexists i \in dom(w).\ SI(w, i) = A_{l+1} \wedge \forall j > i.\ SI(w, j) \leq B_l$

We can construct *Rpo2dfa* to check each of the above properties

The Automata and its Hierarchy

The Recursion Hierarchy

The languages definable by *Rpo2dfa*[k] forms a hierarchy
$Rpo2dfa[k] \subsetneq Rpo2dfa[k+1]$

- *po2dfa* are expressively equivalent to the level $\Delta_2[<]$ of the alternation hierarchy [STV01, TW98].
- For every $Rpo2dfa[k]$, we may construct language-equivalent $\Sigma_{k+1}[<]$ and $\Pi_{k+1}[<]$ sentences.
- Hence, we are able to embed $Rpo2dfa[k]$ within the level $\Delta_{k+1}[<]$ of the alternation hierarchy.
- It can also be shown that the recursion hierarchy is strict: Bounded buffer problem separates these levels

A temporal logic for *Rpo2dfa*

Recursive Temporal Logic ($TL[X_\phi, Y_\phi]$) with the *recursive* and *deterministic Next* and *Prev* modalities.

Syntax

$\phi := \top \mid a \mid X_\phi\phi \mid Y_\phi\phi \mid \phi \vee \phi \mid \neg\phi$

Theorem: There exists a constructive equivalence between $TL[X_\phi, Y_\phi]$ and rpotdfa: For every $TL[X_\phi, Y_\phi]$ formula of level k we may construct a language-equivalent $Rpo2dfa[k]$ and vice versa.

The Limit

Theorem: For every LTL formula, we may construct a language-equivalent $TL[X_\phi, Y_\phi]$ formula.

$$\bigcup_k Rpo2dfa[k] \quad \equiv \quad LTL \quad \equiv \quad FO$$

However, the recursion hierarchy is distinct from the Until-since hierarchy and the dot-depth hirarchy

Some Related Work

- The logic $TL[X_\phi, Y_\phi]$ was defined by Kroger [Krö84], with "at-next" and "at-prev" modalities and showed equivalence to LTL.
- In [BT04], Borchert characterizes the logic, using weakly-iterated block products of the variety DA.
- [Bor04] defines the "at-hierarchy", based on the nesting depth of "at"-modalities and shows that the hierarchy is strict.
  - Level $\Sigma_2[<]$ intersects with all levels of the at-hierarchy.
  - Level k of the at-hierarchy lies strictly below $\Delta_{k+1}[<]$ for every k.
  - The relation between at-hierarchy and US hierarchy was posed as an open question.

The Missing Piece

Relation between US hierarchy and recursion hierarchy:
The unary F and P modalities of *LTL* are indeed “for free” i.e. they do not result in increase in recursion depth of the corresponding *Rpo2dfa*.

Theorem

A given *LTL* formula ϕ may be expressed using a language-equivalent *Rpo2dfa* whose recursion depth is equal to the modal depth of only $\mathcal{U}$ and $\mathcal{S}$ operators of ϕ.

Relies on our conversion from $TL[F, P]$ to *po2dfa* [PS14, Sha12]

A sublogic of $TL[X_\phi, Y_\phi]$

Syntax of $TL^+[X_\phi, Y_\phi]$

$$\psi \;:=\; a \;|\; \phi \;|\; \psi \vee \psi \;|\; \neg\psi$$

where $a \in \Sigma$ and ϕ is of the form

$$\phi \;:= \top \;|\; SP\phi \;|\; EP\phi \;|\; X_\psi\phi \;|\; Y_\psi\phi$$

This "small" restriction brings down the expressiveness of the logic to $\Delta_2[<]$.

Summary

- *Rpo2dfa* and the recursion hierarchy define an alternative automaton-characterization and hierarchy for FO-definable languages
- It has a matching temporal logic and weakly iterated block products of the variety DA
- *Rpo2dfa* are a subclass of recursive state machines, and comparable with alternating automata
- The recursion hierarchy grows “faster” than the US-hierarchy but “slower” than the alternation hierarchy for FO over finite words
- Level k of the US hierarchy can be embedded within level $k+1$ of the recursion hierarchy
- Level k of the recursion hierarchy can be embeded within level Δ_{k+1} of the $FO[<]$ alternation hierarchy
- Complexity of word-membership, satisifability, language-emptiness needs to be explored
- How can we “flatten” these automata?

References I

Bernd Borchert.
The dot-depth hierarchy vs. iterated block products of da, 2004.

Bernd Borchert and Pascal Tesson.
The atnext/atprevious hierarchy on the starfree languages, 2004.

Fred Kröger.
A generalized nexttime operator in temporal logic.
J. Comput. Syst. Sci., 29(1):80–98, 1984.

Paritosh K. Pandya and Simoni S. Shah.
Deterministic logics for ul.
CoRR, abs/1401.2714, 2014.

Paritosh Pandya and Simoni S. Shah.
Recursive *po2dfa*: Hierarchical automata for fo-definable languages (manuscript), 2015.

Simoni S. Shah.
Unambiguity and Timed Languages:Automata, Logics, Expressiveness (Submitted).
PhD thesis, TIFR, Mumbai, 2012.

Thomas Schwentick, Denis Thérien, and Heribert Vollmer.
Partially-ordered two-way automata: A new characterization of *DA*.
In *Developments in Language Theory*, pages 239–250, 2001.

References III

Denis Thérien and Thomas Wilke.
Over words, two variables are as powerful as one quantifier alternation.
In *STOC*, pages 234–240, 1998.

THANK YOU!