Büchi’s Logical Characterisation of Regular Languages

Deepak D’Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

06 September 2012
Outline

1. First-Order Logic of \((\mathbb{N}, <)\)
2. The logic \(\text{MSO}(A)\)
3. Proof of Büchi’s theorem
First-Order Logic of $\langle \mathbb{N}, < \rangle$  

**Background**

- Büchi’s motivation: Decision procedure for deciding truth of first-order logic statements about natural numbers and their ordering. Eg.

\[ \forall x \exists y (x < y). \]

- Used finite-state automata to give decision procedure.

- By-product: a logical characterisation of regular languages.

**Theorem (Büchi 1960)**

$L$ is regular iff $L$ can be described in Monadic-Second Order Logic.
First-Order logic of \((\mathbb{N}, <)\).

- Interpreted over \(\mathbb{N} = \{0, 1, 2, 3, \ldots\}\).
- What you can say:
  
  \[ x < y, \quad \exists x \varphi, \quad \forall x \varphi, \quad \neg, \land, \lor. \]

- Examples:
  1. \(\forall x \exists y (x < y)\).
First-Order logic of \((\mathbb{N}, <)\).

- Interpreted over \(\mathbb{N} = \{0, 1, 2, 3, \ldots\}\).
- What you can say:

\[ x < y, \ \exists x \phi, \ \forall x \phi, \ \neg, \land, \lor. \]

- Examples:
  1. \(\forall x \exists y (x < y)\).
  2. \(\forall x \exists y (y < x)\).

Question: Is there an algorithm to decide if a given FO\((\mathbb{N}, <)\) sentence is true or not?
First-Order logic of \((\mathbb{N}, <)\).

- Interpreted over \(\mathbb{N} = \{0, 1, 2, 3, \ldots\}\).
- What you can say:
  \[ x < y, \; \exists x \varphi, \; \forall x \varphi, \; \neg, \land, \lor. \]
- Examples:
  1. \(\forall x \exists y (x < y)\).
  2. \(\forall x \exists y (y < x)\).
  3. \(\exists x (\forall y (y \leq x))\).
First-Order Logic of \((\mathbb{N}, <)\).

- Interpreted over \(\mathbb{N} = \{0, 1, 2, 3, \ldots\}\).
- What you can say:

  \[ x < y, \quad \exists x \varphi, \quad \forall x \varphi, \quad \neg, \land, \lor. \]

- Examples:
  1. \( \forall x \exists y (x < y) \).
  2. \( \forall x \exists y (y < x) \).
  3. \( \exists x (\forall y (y \leq x)) \).
  4. \( \forall x \forall y ((x < y) \implies \exists z (x < z < y)) \).

Question: Is there an algorithm to decide if a given FO\((\mathbb{N}, <)\) sentence is true or not?
First-Order logic of \((\mathbb{N}, <)\).

- Interpreted over \(\mathbb{N} = \{0, 1, 2, 3, \ldots\}\).
- What you can say:
  \[ x < y, \exists x \varphi, \forall x \varphi, \neg, \land, \lor. \]

- Examples:
  1. \(\forall x \exists y (x < y)\).
  2. \(\forall x \exists y (y < x)\).
  3. \(\exists x (\forall y (y \leq x))\).
  4. \(\forall x \forall y ((x < y) \implies \exists z (x < z < y))\).

- Question: Is there an algorithm to decide if a given \(\text{FO}(\mathbb{N}, <)\) sentence is true or not?
Monadic Second-Order logic over alphabet $A$: MSO($A$)

- Interpreted over a string $w \in A^*$.
  
  $w = a \ a \ b \ a \ b \ a \ b \ a \ b$
  
  $0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8$

- Domain is set of positions in $w$: $\{0, 1, 2, \ldots, |w| - 1\}$.

- “$<$” is interpreted as usual $<$ over numbers.

- What we can say in the logic:
  
  - $Q_a(x)$: “Position $x$ is labelled $a$”.
  - $x < y$: “Position $x$ is strictly less than position $y$”.
  - $\exists x \varphi$: “There exists a position $x$ ...”
  - $\forall x \varphi$: “For all positions $x$ ...”
  - $\exists X \varphi$: “There exists a set of positions $X$ ...”
  - $\forall X \varphi$: “For all sets of positions $X$ ...”
  - $x \in X$: “Position $x$ belongs to the set of positions $X$”. 
Example $\text{MSO}(\{a, b\})$ formulas

What language do the sentences below define?

1. $\exists x (\neg \exists y (y < x) \land Q_a(x))$.
2. $\exists y (\neg \exists x (y < x) \land Q_b(y))$.
3. $\exists x \exists y \exists z (\text{succ}(x, y) \land \text{succ}(y, z) \land \text{last}(z) \land (Q_b(x)))$.

Give sentences that describe the following languages:

1. Every $a$ is immediately followed by a $b$.
2. Strings of odd length.
MSO sentence for strings of odd length

Language $L \subseteq \{a, b\}^*$ of strings of odd length.

\[
\exists x_e \exists x_o (\exists x (x \in X_e) \land (\forall x ((x \in X_e \implies \neg x \in X_o) \land (x \in X_o \implies \neg x \in X_e) \land (x \in X_e \lor x \in X_o) \land (\text{zero}(x) \implies x \in X_e) \land (\forall y ((x \in X_e \land \text{succ}(x, y)) \implies y \in X_o)) \land (\forall y ((x \in X_o \land \text{succ}(x, y)) \implies y \in X_e)) \land (\text{last}(x) \implies x \in X_e))).
\]
A First-Order Logic usually has a **signature** comprising the constants, and function/relation symbols. Eg. \((0, <, +)\).

**Terms** are expressions built out of the constants and variables and function symbols. Eg. \(0, x + y, (x + y) + 0\). They are interpreted as elements of the domain of interpretation.

**Atomic formulas** are obtained using the relation symbols on terms of the logic. Eg. \(x < y, x = 0 + y, x + y < 0\).

**Formulas** are obtained from atomic formulas using boolean operators, and existential quantification \((\exists x)\) and universal quantification \((\forall x)\). Eg. \(\neg(x < y), (x < 0) \land (x = y), \exists x(\forall y(x < y) \land (z < x))\).
Given a “structure” (i.e. a domain, a concrete interpretation for each constant and function/relation symbol) and an assignment for variables to values in the domain) to interpret the formulas in, each formula is either true or false.

A formula is called a **sentence** if it has no free (unquantified) variables.
In **Second-Order** logic, one allows quantification over relations over the domain (not just elements of the domain). Eg:

\[ \exists R^2(R^2(x, y) \implies x < y). \]

In **Monadic** second-order logic, one allows quantification over monadic relations (i.e. relations of arity one, or subsets of the domain). Eg:

\[ \exists X(x \in X \implies 0 < x). \]
An interpretation for the logic will be a pair \((w, \mathcal{I})\) where \(w \in A^*\) and \(\mathcal{I}\) is an assignment of individual variables to a position in \(w\), and set variables to a set of positions of \(w\).

\[\mathcal{I} : Var \rightarrow pos(w) \cup 2^{pos(w)}.\]

\(\mathcal{I}[i/x]\) denotes the assignment which maps \(x\) to \(i\) and agrees with \(\mathcal{I}\) on all other individual and set variables.

Similarly for \(\mathcal{I}[S/X]\).
Formal Semantics of MSO

The satisfaction relation $w, \mathcal{I} \models \varphi$ is given by:

- $w, \mathcal{I} \models Q_a(x)$ if and only if $w(\mathcal{I}(x)) = a$
- $w, \mathcal{I} \models x < y$ if and only if $\mathcal{I}(x) < \mathcal{I}(y)$
- $w, \mathcal{I} \models x \in X$ if and only if $\mathcal{I}(x) \in \mathcal{I}(X)$
- $w, \mathcal{I} \models \neg \varphi$ if and only if $w, \mathcal{I} \not\models \varphi$
- $w, \mathcal{I} \models \varphi \lor \varphi'$ if and only if $w, \mathcal{I} \models \varphi$ or $w, \mathcal{I} \models \varphi'$
- $w, \mathcal{I} \models \exists x \varphi$ if and only if there exists $i \in pos(w)$ such that $w, \mathcal{I}[i/x] \models \varphi$
- $w, \mathcal{I} \models \exists X \varphi$ if and only if there exists $S \subseteq pos(w)$ such that $w, \mathcal{I}[S/X] \models \varphi$
**MSO sentences**

- A **sentence** is a formula with no free variables.
- For example, $\exists X (y \in X \implies 0 < y)$ is not a sentence since $y$ occurs free.
- $\exists X (0 \in X \implies \exists y (0 < y \land y \in X))$ is a sentence.
- If $\varphi$ is a sentence, then we don't need an interpretation for variables to say if $\varphi$ is true or false of a given word $w$:
  \[ w \models \varphi. \]
- For a sentence $\varphi$, we can define the language of words that satisfy $\varphi$:
  \[ L(\varphi) = \{ w \in A^* \mid w \models \varphi \}. \]
Languages definable by MSO

- We say that a language $L \subseteq A^*$ is definable in MSO($A$) if there is a sentence $\varphi$ in MSO($A$) such that $L(\varphi) = L$.

**Theorem (Büchi 1960)**

$L \subseteq A^*$ is regular iff $L$ is definable in MSO($A$).
From automata to MSO sentence

- Let $L \subseteq A^*$ be regular. Let $A = (Q, s, \delta, F)$ be a DFA for $L$.
- To show $L$ is definable in $\text{MSO}(A)$.
- Idea: Construct a sentence $\varphi_A$ describing an accepting run of $A$ on a given word.
  That is: $\varphi_A$ is true over a given word $w$ precisely when $A$ has an accepting run on $w$.

Let $Q = \{q_1, \ldots, q_n\}$, with $q_1 = s$.
Define $\varphi_A$ as

$$
\exists X_1 \cdots \exists X_n (\forall x ( 
(\bigwedge_{i \neq j} (x \in X_i \implies \neg x \in X_j) \land \bigvee_i x \in X_i) \land 
(zero(x) \implies x \in X_1) \land 
(\bigwedge_{a \in A, \ i,j \in \{1, \ldots, n\}, \ \delta(q_i,a) = q_j} ((x \in X_i \land Q_a(x) \land \neg last(x)) \implies 
\exists y (\text{succ}(x, y) \land y \in X_j))) \land 
(last(x) \implies \bigvee_{a \in A, \ \delta(q_i,a) \in F} (Q_a(x) \land x \in X_i))))).
$$
Example

Consider language $L \subseteq \{a, b\}^*$ of strings of even length.

DFA $\mathcal{A}$ for $L$:

\[
\begin{array}{cccccccc}
a & a & b & a & b & a & b & a & b \\
X_e & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
X_o & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\
\end{array}
\]

$\varphi \mathcal{A}$:

\[
\exists X_e \exists X_o (\forall x ( (x \in X_e \implies \neg x \in X_o) \land (x \in X_o \implies \neg x \in X_e) \land (x \in X_e \lor x \in X_o) \land \\
(zero(x) \implies x \in X_e) \land ((x \in X_e \land Q_a(x) \land \neg last(x)) \implies \exists y (succ(x, y) \land y \in X_o)) \land \\
((x \in X_e \land Q_b(x) \land \neg last(x)) \implies \exists y (succ(x, y) \land y \in X_e)) \land \\
((x \in X_o \land Q_a(x) \land \neg last(x)) \implies \exists y (succ(x, y) \land y \in X_e)) \land \\
((x \in X_o \land Q_b(x) \land \neg last(x)) \implies \exists y (succ(x, y) \land y \in X_e)) \land \\
(last(x) \implies ((Q_a(x) \land x \in X_o) \lor (Q_b(x) \land x \in X_o))))).
\]
From MSO sentence to automaton

- Idea: Inductively describe the language of extended models of a given MSO formula $\varphi$ by an automaton $A_\varphi$.
- Extended models wrt set of first-order and second-order variables $T = \{x_1, \ldots, x_m, X_1, \ldots, X_n\}$: $(w, I)$
- Can be represented as a word over $A \times \{0, 1\}^{m+n}$.

For example above extended word satisfies the formula

$$Q_a(x_1) \land (x_2 \in X_1).$$
Inductive construction of $A^T_{\varphi}$.

- If $\varphi$ is a formula whose free variables are in $T$, then we have the notion of whether $w' \models \varphi$ based on whether the $(w, I)$ encoded by $w'$ satisfies $\varphi$ or not.
- Let the set of valid extended words wrt $T$ be $valid^T(A)$.
- We can define an automaton $A^T_{val}$ which accepts this set.
- Claim: with every formula $\varphi$ in MSO($A$), and any finite set of variables $T$ containing at least the free variables of $\varphi$, we can construct an automaton $A^T_{\varphi}$ which accepts the language $L^T(\varphi)$.
- Proof: by induction on structure of $\varphi$.

$Q_a(x), \ x < y, \ x \in Y, \ \neg \varphi, \ \varphi \lor \psi, \ \exists x \varphi, \ \exists X \varphi.$
Interpreted over $\mathbb{N} = \{0, 1, 2, 3, \ldots\}$.

What you can say:

$x < y$, $\exists x \varphi$, $\forall x \varphi$, $\neg$, $\land$, $\lor$.

Examples:

1. $\forall x \exists y (x < y)$.
2. $\forall x \exists y (y < x)$.
3. $\exists x (\forall y (y \leq x))$.
4. $\forall x \forall y ((x < y) \implies \exists z (x < z < y))$.

Question: Is there an algorithm to decide if a given FO($\mathbb{N}, <$) sentence is true or not?
Büchi’s decision procedure for MSO(\(\mathbb{N}, <\))

- Büchi considered finite automata over infinite strings (so called \(\omega\)-automata).
- An infinite word is accepted if there is a run of the automaton on it that visits a final state infinitely often.
- Büchi showed that \(\omega\)-automata have similar properties to classical automata: are closed under boolean operations, projection, and can be effectively checked for emptiness.
- MSO characterisation works similarly for \(\omega\)-automata as well.
- Given a sentence \(\varphi\) in MSO(\(\mathbb{N}, <\)) we can now view it as an MSO(\(\{a\}\)) sentence.
- Construct an \(\omega\)-automaton \(A_\varphi\) that accepts precisely the words that satisfy \(\varphi\).
- Check if \(L(A_\varphi)\) is non-empty.
- If non-empty say “Yes, \(\varphi\) is true”, else say “No, it is not true.”
Summary

- We saw another characterisation of the class of regular languages, this time via logic:

  **Theorem (Büchi 1960)**
  
  \[ L \subseteq A^* \text{ is regular iff } L \text{ is definable in } \text{MSO}(A). \]

- We saw an application of automata theory to solve a decision procedure in logic:

  **Theorem (Büchi 1960)**
  
  *The Monadic Second-Order (MSO) logic of \((\mathbb{N}, <)\) is decidable.*