Languages defined by a first order logic over an alphabet

Ishan Agarwal Sayantan Khan

Indian Institute of Science

Friday 16th September, 2016

Outline

First order logic over an alphabet Σ

- Sentences in this logic assign True/False values to elements of Σ^*.
- The atomic predicates in this logic are $<$, which is a binary predicate, and Q_k for each $k \in \Sigma$, which is a unary predicate.
- One can make larger formulae using the boolean connectives, namely $\neg$, $\wedge$, and $\vee$.
- One can also make formulae of the form $\forall x \psi$ or $\exists x \psi$, where ψ is a first order formula, and x is a variable in the domain, i.e. a subset of natural numbers.

Interpreting the first order logic over Σ^*

- If $w \in \Sigma^*$, then the domain over which the variables take value is the set $\{0, 1, \ldots, |w| - 1\}$.
- $Q_a(x)$ is true if the letter at position x is a (the first letter is at position 0).
- $x < y$ is true if $x < y$ when x and y are interpreted as natural numbers.
- $\forall x \psi$ is true if $\psi(x)$ is true for all $x \in \{0, 1, \ldots, |w| - 1\}$. $\exists x \psi$ is interpreted in an analogous manner.
- For a given sentence ψ, the subset of Σ^* for which the sentence evaluates to True is the language defined by ψ.

Theorem (Corollary of Büchi's theorem)

A language defined by a first order logical sentence is regular.

Counter free languages and automata

- A DFA has a counter if there exist states $q_0, q_1, \ldots q_{n-1}$, where $n \geq 2$, such that for some word $w \in \Sigma^*$, $\widehat{\delta}(q_i, w) = q_{i+1}$ for $0 \leq i \leq n-2$ and $\widehat{\delta}(q_{n-1}, w) = q_0$.
- A regular language is counter free if its minimal DFA does not have a counter.

Temporal logic over an alphabet Σ

- Atomic predicates in this logic are $\top$ (True), $\bot$ (False), and a for each $a \in \Sigma$.
- Larger formulae are made using the boolean connectives $\neg$, $\wedge$, and $\vee$.
- One can also use *temporal modalities* like **X** (next), **F** (eventually), and **U** (until) to get formulae of the form $\mathbf{X}\psi$, $\mathbf{F}\psi$, or $\phi\mathbf{U}\psi$.

Interpreting temporal logic over Σ^*

- $\top$ is satisfied by all words in Σ^* and $\perp$ is satisfied by no word in Σ^*.
- Given a word $u \in \Sigma^*$, $u(0)$ is the first letter in the word. The atomic predicate a is satisfied by u if $u(0) = a$.
- Given a word u, $u(i, *)$ is the suffix of u obtained by truncating the first i letters. A word u satisfies $\mathbf{X}\psi$ if $u(1, *)$ satisfies ψ.
- Given a word u, u satisfies $\mathbf{F}\psi$ if for some $i > 0$, $u(i, *)$ satisfies ψ.
- $\phi\mathbf{U}\psi$ is satisfied by a word u if there exists $0 < i < |u|$ such that for all $0 < j < i$, $u(j, *)$ satisfies ϕ and $u(i, *)$ satisfies ψ.

The Equivalence Theorem

Theorem (CF $\equiv$ FO $\equiv$ TL)

Given a language L over an alphabet Σ, L is counter free iff L is defined by a sentence in first order logic over Σ, and L is defined by a sentence in first order logic iff it is defined by a sentence in temporal logic.

Outline of proof

- We will show that a language defined by a sentence in TL can be defined by a sentence in FOL. Then we'll show a language defined by an FOL sentence is counter free. And finally, we'll show a counter free language can be defined by a sentence in TL.
- To show TL $\implies$ FOL, we'll inductively define a way of translating a TL sentence to an FOL sentence that defines the same language.
- To show FOL $\implies$ CF, we'll adapt the proof of Büchi's theorem, and show that if we restrict ourselves to first order quantifiers, we indeed get a counter free automaton.
- To show CF $\implies$ TL, we'll induct on $|Q|$, where Q is the state space of DFA for the language, and also induct on $|\Sigma|$, where Σ is the alphabet.

Translating TL atomic predicates to FOL

- We can translate $\top$ into FOL by writing a tautology: $\forall x(x = x)$. Similarly, $\perp$ gets translated to $\neg\forall x(x = x)$.
- For $a \in \Sigma$, the predicate a in TL is satisfied by a word if the first letter is a. Translating that into FOL gives us $\exists x(\neg\exists y(y < x) \wedge Q_a(x))$.

Translating $\mathbf{X}\psi$ to FOL

- To translate $\mathbf{X}\psi$, we need to come up with an FOL sentence that satisfies a word u iff the FOL translate χ of ψ is satisfied by the word $u(1,*)$. We need to modify χ somehow such that for all quantifiers in χ, the domain is $\{1, 2, \ldots, |u| - 1\}$ instead of $\{0, 1, \ldots, |u| - 1\}$.
- Consider the following FOL sentence: $\exists f(\neg\exists y(y < f) \wedge \chi')$, where χ' is obtained by modifying each quantifier in χ in the following manner:
  - $\exists x\psi$ is replaced by $\exists x((x > f) \wedge \psi')$.
  - $\forall x\psi$ is replaced by $\forall x((x \leq f) \vee \psi')$.
- We'll call this transformation of χ to χ' as *suffixing* χ by f.

Translating $\mathbf{F}\psi$ to FOL

- Given a first order translation χ of the temporal logic formula ψ, we write $\mathbf{F}\psi$ in a manner similar to the translation of $\mathbf{X}\psi$.
- The sentence $\exists f(\chi')$, where χ' is χ suffixed by f.

Translating $\phi \mathbf{U} \psi$ to FOL

- A similar technique can be used to translate $\phi \mathbf{U} \psi$ to FOL.
- Given TL formulae ϕ and ψ, with their first order translations being ρ and χ respectively, the translation for $\phi \mathbf{U} \psi$ is

$$\exists f((\forall g(g \geq f) \vee \rho') \wedge \chi')$$

Here, ρ' is obtained by suffixing ρ by g, and χ' is obtained by suffixing χ by f.

Showing FOL $\implies$ CF

- The automata corresponding to the atomic predicates $x < y$, and $Q_a(x)$ are counter free.
- Counter free languages are closed under finite union, intersection, and complementation. This shows if the automaton for ψ and ϕ is counter free, then the automatons for $\psi \wedge \phi$, $\psi \vee \phi$, and $\neg\psi$ are also counter free.
- All we need to show now is that the automaton for $\exists x\psi$ is counter free if the automaton for ψ is counter free. The analogous result for $\forall x\psi$ will follow because $\forall x\psi \iff \neg\exists x\neg\psi$.

Showing automaton for $\exists x \psi$ is counter free

- In general, counter free languages are not closed under geometric projections.
- However, when constructing automaton for $\exists x \psi$, the row being projected away has the property that it has *exactly* one 1, and the other letters are 0.
- Given a DFA D for ψ, we construct an NFA for $\exists x \psi$ by taking two copies D_1 and D_2 of D, and keeping transitions within D_1 to be the transition corresponding to $x = 0$, and do the same for D_2. We keep a transition from D_1 to D_2 which corresponds to the transition that happens when $x = 1$. The start state of the NFA is the start state of D_1, and the final states are the final states of D_2.

Example of NFA construction for $\exists x \psi$

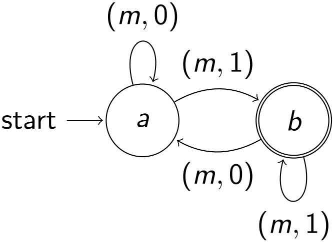

Figure: DFA for some predicate ψ over the alphabet $\{m\} \times \{0, 1\}$.

Example of NFA construction for $\exists x \psi$

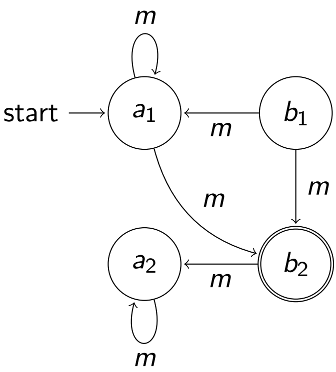

Figure: NFA for $\exists x \psi$ obtained by projecting away the x row.

Showing automaton for $\exists x \psi$ is counter free

We need to show if the automaton for ψ is counter free, then the NFA obtained for $\exists x \psi$ by the described method is also counter free. The proof follows from the following lemma:

Lemma

A language L is not counter free iff there exist words u, v, and w, and an increasing sequence of natural numbers $k_1, k_2, \ldots$ such that $uv^{k_i}w$ belongs to L for odd i and does not belong to L for even i.

Using pre-automata

- A pre-automaton is an automaton without specially distinguished start and final states.
- Let Q be the set of states of a pre-automaton A.A transformation of a string u, relative to the pre-automaton A, is denoted by u^A and is a map from Q to Q given by $u^A(q) = \widehat{\delta}(q, u)$.
- Define $S_A = \{u^A : u \in \Sigma^*\}$. This is called the transformation semi-group of A.
- We also need some notion of a pre-automaton accepting a language. We define $L_\alpha^A = \{u \in \Sigma^+ : u^A = \alpha\}$. Here α is a map from Q to Q.
- We will now show that for all A which arise from counter free automata, and all $\alpha \in S_A$, any language in L_α^A is expressible in temporal logic. This is enough to show the required equivalence.

Proof by induction

- We first show that if α is a surjection then it must be the identity if A is counter free.
- For single state automata we are done.
- We now show the result using automata with same state number but smaller alphabet (L^B_β), as well as assuming the result for lower state number but a much larger alphabet size (L^C_γ).
- The proof proceeds by induction on both $|Q|$ and $|\Sigma|$.

Lemmas

- We know by induction hypothesis that for all β in S_B and all γ in S_C, L^B_β and L^C_γ are expressible in temporal logic. We now write L^A_α in terms of unions and intersections of L^B_β, L^C_γ, Σ^*, T^* etc. We can show by induction that these unions and intersections are all expressible in temporal logic.
- Thus we use the fact that the terms in which we finally express L^A_α are indeed Temporal logic expressible but these are easy to show by straight-forward inductions.

Conclusion

- We have shown that TL implies FOL implies CF implies TL. Thus we have proved the equivalence of all three classes.
- Thus we can use TL in situations where it provides a more intuitive way of proceeding without any loss of expressive power from FOL.
- Further we see that while dealing with statements in FOL or in FOL fragments of other logics, we can safely assume we have a counter free automata for any regular language as counters do not add any expressive power under these conditions.
- In some sense we see that allowing counters in automata is a trade-off for gaining expressive power, for example if we have an MSO sentence that is not in the first order fragment it cannot be represented by a CFA.

For Further Reading

Büchi, J.R.
On a decision method in restricted second order arithmetic
Proc. International Congress on Logic, Method, and Philosophy of Science

Thomas Wilke
Classifying Discrete Temporal Properties
Lecture Notes in Computer Science, Volume 1563, pp 32-46