Learning $k$-reversible languages

Acknowledgements

• Laurent Miclet, Jose Oncina and Tim Oates for previous versions of these slides.
• Rafael Carrasco, Paco Casacuberta, Rémi Eyraud, Philippe Ezequel, Henning Fernau, Thierry Murgue, Franck Thollard, Enrique Vidal, Frédéric Tantini, ...
• List is necessarily incomplete. Excuses to those that have been forgotten.

http://eurise.univ-st-etienne.fr/~cdlh/slides

Outline

1. Introduction
2. Definitions
3. The algorithm
4. Properties

1. The problem

• Gold: identification from positive data only is impossible for super-finite classes.
• The problem thus is over-generalisation.

Avoid

• Accumulation points;
• Languages that do not have tell-tale sets.

2. The $k$-reversible languages

• The class was proposed by Angluin (1982).
• The class is identifiable in the limit from text.
• The class is composed by regular languages that can be accepted by a DFA such that its reverse is deterministic with a look-ahead of $k.$
Let $A=(\Sigma, Q, I, F, \delta)$ be a NFA, we denote by $A^r=(\Sigma, Q, F, I, \delta^r)$ the reversal automaton with:

$$\delta^r(q,a) = \{q' \in Q : q \in \delta(q',a)\}$$

Some definitions

- $u$ is a $k$-successor of $q$ if $|u|=k$ and $\delta(q,u) \neq \emptyset$.
- $u$ is a $k$-predecessor of $q$ if $|u|=k$ and $\delta^r(q,u) \neq \emptyset$.
- $\lambda$ is 0-successor and 0-predecessor of any state.

A NFA is deterministic with look-ahead $k$ iff $\forall q, q' \in Q: q \neq q'$

$$(q, q' \in I) \lor (q, q' \in \delta(q'',a))$$

$$\implies (u \text{ is a } k\text{-successor of } q) \land (v \text{ is a } k\text{-successor of } q') \implies u \neq v$$

Prohibited:
Note

• You must have intersection free successor sets only for those states that have a non determinism issue!

Example

This automaton is not deterministic with look-ahead 1 but is deterministic with look-ahead 2. The fact that states 1 and 2 have common successors is not a problem.

K-reversible automata

• A is k-reversible if A is deterministic and A' is deterministic with look-ahead k.

• Example

3 Learning k-reversible automata

• Key idea: the order in which the merges are performed does not matter!

• Just merge states that do not comply with the conditions for k-reversibility.

Violation of K-reversibility

• Two states q, q’ violate the k-reversibility condition if
  - they violate the deterministic condition: q, q’ ∈ δ(q", a);
  or
  - they violate the look-ahead condition:
    • q, q’ ∈ F, ∃ u ∈ Σk: u is k-predecessor of both;
    • ∃ u ∈ Σk, δ(q, a) = δ(q’, a) and u is k-predecessor of both q and q’.

K-RL Algorithm (α_{K,RL})

Data: k ∈ N, X text sample
A=PTA(X)
while ∃ q, q’ k-reversibility violators do
  A=merge(A, q, q’)

3 Learning K-reversible automata
**K-RL algorithm (a\_k-RL) (with partitions)**

Data: \( k \in \mathbb{N} \), \( X \) text sample

\[
A_0 = \text{PTA}(X) \\
\pi = \{ \{ q \} : q \in Q \} \\
\text{while} \ \exists B, B' \in \pi \ \text{\( k \)-reversibility violators do} \\
\pi = \pi - B - B' \cup \{ B \cup B' \} \\
A = A_0 / \pi
\]

Let \( X = \{ a, aa, abba, abbbba \} \)

\( k = 2 \)

This automaton is 2 reversible. Note that with \( k=1 \) more merges are possible

**Properties (1)**

- \( \forall k \geq 0, \ \forall X, \ a_{k-RL}(X) \) is a \( k \)-reversible language.
- \( L(a_{k-RL}(X)) \) is the smallest \( k \)-reversible language that contains \( X \).
- The class \( k-RL \) is identifiable in the limit from text.

**Properties (2)**

- A regular language is \( k \)-reversible iff
  \[
  \forall u_1, u_2 \in \Sigma^*, \ \forall v \in \Sigma^k, \\
  \exists w \in \Sigma^*: u_1vw \in L \land u_2vw \in L \\
  \Rightarrow (u_1v)^{-1}L = (u_2v)^{-1}L
  \]
  (any two strings who have a common suffix of length \( k \) can be ended, then the strings are Nerode-equivalent)
Properties (3)

- \( L(\alpha_{k-RL}(X)) \subseteq L(\alpha_{[k-1]-RL}(X)) \)
- Remember we also had: \( L(\alpha_{k-TSS}(X)) \subseteq L(\alpha_{[k-1]-TSS}(X)) \)

Properties (4)

- The time complexity is in \( O(k \| X \|^3) \).
- The space complexity is in \( O(\| X \|) \).
- The algorithm can be made incremental.

Polynomial aspects

- Polynomial update time;
- Polynomial characteristic samples?
- Polynomial number of mind changes?
- Polynomial number of implicit prediction errors?

Extensions

- Sakakibara built an extension for context-free grammars whose tree language is \( k \)-reversible.
- Marion & Besombes propose an extension to tree languages.
- Different authors propose to learn these automata and then estimate the probabilities as an alternative to learning stochastic automata.

Exercises

- Construct a language \( L \) that is not \( k \)-reversible, \( \forall k \geq 0 \).
- Prove that the class of \( k \)-reversible languages is not identifiable from text.
- Run \( \alpha_{k-RL} \) on \( X=\{aa,aba,abb,abaaba,baaba\} \) for \( k=0,1,2,3 \).