

ALEX: Auto-completion Learning for XML

Editor

Serge Abiteboul

Yael Amsterdamer

Pierre Senellart

Tova Milo



Problem

XML documents are often edited manually

- Examples: XHTML, Docbook,...
- Many editors are available
- Still, manually editing XML docs can be hard

Auto-completion can help!

How to choose completion suggestions?

- Main idea: **learn probabilistically** the completions from an XML corpus!
- Of the same user, or "similar" ones
- Example: XML homepage

Challenges

- Intricate structure
- Integrity constraints
- Online analysis is required

Problem statement

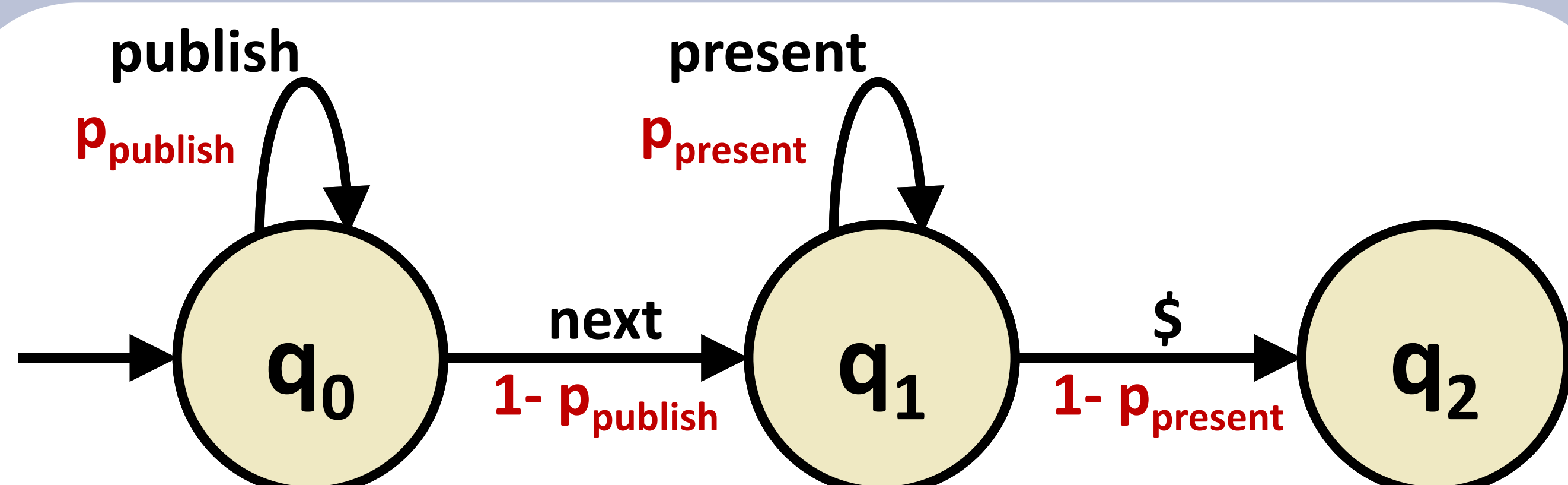
Find the most likely XML document part given

- A model
- Integrity (key, inclusion, domain) constraints
- A partial document

Do it repeatedly, and fast!

Model

- A **generative model** based on **Tree Automata**
- DFAs describe possible children of a node
- **Probabilities** annotate transitions



Complexity Results

- Checking for the existence of a valid continuation is already **NP-complete** in the schema size
- We can find **most likely continuation** in time **exponential** in the schema size, **polynomial** in partial document and output size
- Schema size is typically small!

Techniques

- Algorithm to **learn a model** instance that is provably the **most likely estimator** for a given corpus

NP-complete in the schema size,
polynomial in the corpus size

- **A*-like** algorithm to find **most likely continuations**
- Reducing the number of examined continuations

Implementation and Optimizations

The Editor is a **plugin to Eclipse IDE** based on **Rinzo**

- Demonstrated with **real data** about researchers

Dedicated **optimizations** to allow interactivity

- Pruning options that lead to a "dead end"
- Separate structure and value generation
- Configurable bound of continuation search depth

System Architecture

