Fast Learning of Restricted Regular Expressions and DTDs

Dominik D. Freydenberger[1] Timo Kötzing[2]

[1] Johann-Wolfgang-Goethe-Universität, Frankfurt am Main

[2] Friedrich-Schiller-Universität, Jena

ICDT 2013

Schema Inference

Schema specification

- in principle, XML documents are supposed to have a schema specification
- allows validation
- various tools expect presence of a schema

Schema Inference

Schema specification

- in principle, XML documents are supposed to have a schema specification
- allows validation
- various tools expect presence of a schema

The bitter truth

In most XML documents in the real world, the schema is specified incorrectly or not at all.

Solution: Schema inference

Automatically generate a "good" schema from positive examples.

Schema Inference

Schema specification

- in principle, XML documents are supposed to have a schema specification
- allows validation
- various tools expect presence of a schema

The bitter truth

In most XML documents in the real world, the schema is specified incorrectly or not at all.

Solution: Schema inference

Automatically generate a “good” schema from positive examples.

Why focus on DTDs?

- essential part of learning XSD
- useful for learning RELAX NG
- human readable
- still widely used

A closer look at DTDs

The biggest challenge for learning:

Element type declarations

$\langle$!ELEMENT book (title, author+, dedication?, chapter*)$\rangle$
$\langle$!ELEMENT chapter ((figure|paragraph)*)$\rangle$

- element name
- , list operator
- $+$ one or more
- ? zero or one
- $*$ zero or more
- $|$ choice

A closer look at DTDs

The biggest challenge for learning:

Element type declarations

⟨!ELEMENT book (title, author+, dedication?, chapter∗)⟩
⟨!ELEMENT chapter ((figure|paragraph)∗)⟩

- element name
- , list operator
- + one or more
- ? zero or one
- ∗ zero or more
- | choice

Central observation

every element type declaration is a (deterministic) regular expression

A closer look at DTDs

The biggest challenge for learning:

Element type declarations

$\langle$`!ELEMENT book (title, author+, dedication?, chapter*)`$\rangle$
$\langle$`!ELEMENT chapter ((figure|paragraph)*)`$\rangle$

- element name
- , list operator
- $+$ one or more
- $?$ zero or one
- $*$ zero or more
- $|$ choice

- letter
- concatenation
- Kleene $+$
- union with $\{\varepsilon\}$
- Kleene *
- union

Central observation

every element type declaration is a (deterministic) regular expression

Problem statement

Original problem

Given a finite set of XML documents, find a good DTD.

Problem statement

Original problem

Given a finite set of XML documents, find a good DTD.

Actual problem

Given a finite set of words, find a good regular expression.

Problem statement

Original problem

Given a finite set of XML documents, find a good DTD.

Actual problem

Given a finite set of words, find a good regular expression.

finite set of words
S

$\Longrightarrow$

good regular expression:

Problem statement

Original problem

Given a finite set of XML documents, find a good DTD.

Actual problem

Given a finite set of words, find a good regular expression.

finite set of words
S

$\Longrightarrow$

good regular expression:

- deterministic
- generates superset of S
- must avoid overgeneralization
- should be concise

Problem statement

Original problem

Given a finite set of XML documents, find a good DTD.

Actual problem

Given a finite set of words, find a good regular expression.

finite set of words S $\Longrightarrow$

good regular expression:

- deterministic
- generates superset of S
- must avoid overgeneralization
- should be concise

- setting is similar to Learning in the Limit (Gold 1967)
  aka Gold-style Learning, Explanatory Learning, Inductive Inference
- Gold: impossible for the full class of (det.) regular expressions
- $\Rightarrow$ need good restrictions

Previous work

- Bex, Neven, Schwentick, Tuyls:
  Inference of concise DTDs from XML data. VLDB 2006.
- Bex, Neven, Schwentick, Vansummeren:
  Inference of concise regular expressions and DTDs. ACM TODS 2010.

Previous work

- Bex, Neven, Schwentick, Tuyls:
  Inference of concise DTDs from XML data. VLDB 2006.
- Bex, Neven, Schwentick, Vansummeren:
  Inference of concise regular expressions and DTDs. ACM TODS 2010.

SORE

Single **O**ccurrence **R**egular **E**xpression
Every letter occurs only once in the expression.

Example

$((\mathtt{a} \mid \mathtt{b})^+ c?)^+$
$(\mathtt{a}\,\mathtt{b})^+ c$

Previous work

- Bex, Neven, Schwentick, Tuyls:
  Inference of concise DTDs from XML data. VLDB 2006.
- Bex, Neven, Schwentick, Vansummeren:
  Inference of concise regular expressions and DTDs. ACM TODS 2010.

SORE

Single **O**ccurrence **R**egular **E**xpression
Every letter occurs only once in the expression.

Example

$((\mathtt{a} \mid \mathtt{b})^+ c?)^+$
$(\mathtt{a}\,\mathtt{b})^+ c$

CHARE

Chain **R**egular **E**xpression
SORE, and only a concatenation of chain factors $(\mathtt{a}_1 \mid \cdots \mid \mathtt{a}_n)\circ$, where $\circ \in \{?, ^+, ^*, \epsilon\}$

Example

$(\mathtt{a} \mid \mathtt{b})(\mathtt{c} \mid \mathtt{d})$
$(\mathtt{a} \mid \mathtt{b} \mid \mathtt{c})^+ \mathtt{d}?$

Some Difficulties

Key assumptions of Gold style(-ish) learning

- target language T belongs to the target class $\mathcal{C}$
- S contains sufficient information to identify T

- Bex et al. give algorithms that learn CHAREs or SOREs if these conditions are satisfied

Some Difficulties

Key assumptions of Gold style(-ish) learning

- target language T belongs to the target class $\mathcal{C}$
- S contains sufficient information to identify T

- Bex et al. give algorithms that learn CHAREs or SOREs if these conditions are satisfied
- But what if...
  - ...the target language T does not belong to $\mathcal{C}$?
  - ...the information in the sample S is insufficient?

Some Difficulties

Key assumptions of Gold style(-ish) learning

- target language T belongs to the target class $\mathcal{C}$
- S contains sufficient information to identify T

- Bex et al. give algorithms that learn CHAREs or SOREs if these conditions are satisfied
- But what if...
  - ...the target language T does not belong to $\mathcal{C}$?
  - ...the information in the sample S is insufficient?
- existing algorithms compute generalizations of T...
  ...these might be overgeneralizations

Question

Is there an aesthetic and efficient solution to these problems?

Descriptive Generalization

Good news!

There is a model that addresses these problems: **Descriptive Generalization** (F., Reidenbach; COLT 2010)

similar to Gold-style learning

Descriptive Generalization

Good news!

There is a model that addresses these problems:
Descriptive Generalization
(F., Reidenbach; COLT 2010)

similar to Gold-style learning

Descriptive Representations

Let $\mathcal{R}$ be a set of language representations.
$\delta \in \mathcal{R}$ is **$\mathcal{R}$-descriptive** of a language S if

1. $L(\delta) \supseteq S$, and
2. there is no $\gamma \in \mathcal{R}$ with $L(\delta) \supset L(\gamma) \supseteq S$.

Descriptive Generalization

Good news!

There is a model that addresses these problems:
Descriptive Generalization
(F., Reidenbach; COLT 2010)

similar to Gold-style learning

Descriptive Representations

Let $\mathcal{R}$ be a set of language representations. $\delta \in \mathcal{R}$ is **$\mathcal{R}$-descriptive** of a language S if

1. $L(\delta) \supseteq S$, and
2. there is no $\gamma \in \mathcal{R}$ with $L(\delta) \supset L(\gamma) \supseteq S$.

Descriptive Generalization

Instead of trying to find an exact representation of T, we try to compute a $\delta \in \mathcal{R}$ that is $\mathcal{R}$-descriptive of S.

- Classical $\mathcal{R}$: pattern languages (Angluin 1979)
- We use CHAREs or SOREs as $\mathcal{R}$
  $\Rightarrow$ compute descriptive CHAREs/SOREs

Main Results

Observation

- The algorithms by Bex et al. do not (always) compute descriptive CHAREs/SOREs.
- This already holds for very small alphabets.

Main Results

Observation

- The algorithms by Bex et al. do not (always) compute descriptive CHAREs/SOREs.
- This already holds for very small alphabets.

Our algorithms

Given a sample S, we can compute

- a CHARE-descriptive CHARE in time $O(ln + m)$ (Bex et al.: $O(ln + n^3)$)
- a SORE-descriptive SORE in time $O(ln + mn)$ (Bex et al.: $O(ln + n^5)$)

- l: size of the sample S ($=\sum_{w \in S} |w|$)
- m: number of different 2-factors in S, $m \leq n^2$
- n: size of the alphabet

Our algorithms are more precise and (probably) more efficient.

Practical Examples

We used a prototype implementation to create a few examples.

Test data

- Mondial database
- MEDLINE/PubMed

Why those?

- come with DTDs. . .
- . . . that are non-trivial

Practical Examples

We used a prototype implementation to create a few examples.

Test data

- Mondial database
- MEDLINE/PubMed

Why those?

- come with DTDs. . .
- . . . that are non-trivial

Observations

- Most of the element type declarations in the DTDs are CHAREs,
- all element type declarations are SOREs,
- (mostly) identical expressions are found by our algorithms.
- There are original declarations that are too general (according to the data).

Example 1: island (Mondial)

Original DTD

⟨!ELEMENT island
(name,islands?,located*,area?,elevation?, longitude?,latitude?)⟩

Example 1: island (Mondial)

Original DTD

⟨!ELEMENT island
(name,islands?,located*,area?,elevation?, longitude?,latitude?)⟩

Descriptive CHARE

⟨!ELEMENT island
(name,islands?,located*,area?,elevation?, longitude?,latitude?)⟩

Example 1: island (Mondial)

Original DTD

⟨!ELEMENT island
(name,islands?,located*,area?,elevation?, longitude?,latitude?)⟩

Descriptive CHARE

⟨!ELEMENT island
(name,islands?,located*,area?,elevation?, longitude?,latitude?)⟩

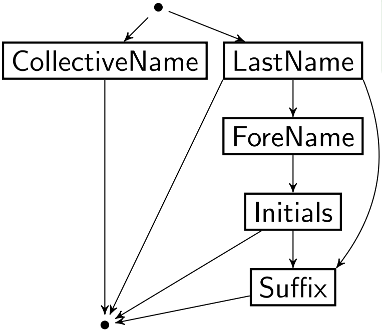
Descriptive SORE

⟨!ELEMENT island
(name,islands?,located*,area?,elevation?,(longitude,latitude)?)⟩

Observation

- SOREs can model dependencies

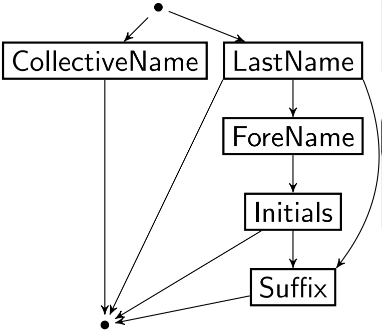
Example 2: author (Medline)



Official DTD

((LastName, ForeName?, Initials?, Suffix?)
| CollectiveName), Identifier*

Example 2: author (Medline)



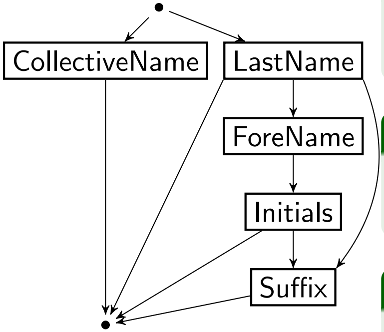
Official DTD

((LastName, ForeName?, Initials?, Suffix?) | CollectiveName), Identifier*

Descriptive CHARE

(LastName | CollectiveName), ForeName?, Initials?, Suffix?

Example 2: author (Medline)



Official DTD

((LastName, ForeName?, Initials?, Suffix?)
| CollectiveName), Identifier*

Descriptive CHARE

(LastName | CollectiveName), ForeName?,
Initials?, Suffix?

Descriptive SORE

(LastName, (ForeName, Initials)?, Suffix?)
| CollectiveName

Observation

- CHAREs have to serialize choice

Example 3: MedlineCitation (Part 1/2)

MedlineCitation from the MEDLINE files

Official DTD

⟨!ELEMENT MedlineCitation (PMID, DateCreated, DateCompleted?, DateRevised?, Article, MedlineJournalInfo, ChemicalList?, SupplMeshList?, CitationSubset*, CommentsCorrectionsList?, GeneSymbolList?, MeshHeadingList?, NumberOfReferences?, PersonalNameSubjectList?, OtherID*, OtherAbstract*, KeywordList*, SpaceFlightMission*, InvestigatorList?, GeneralNote*)⟩

Example 3: MedlineCitation (Part 1/2)

MedlineCitation from the MEDLINE files

Official DTD

⟨!ELEMENT MedlineCitation (PMID, DateCreated, DateCompleted?, DateRevised?, Article, MedlineJournalInfo, ChemicalList?, SupplMeshList?, CitationSubset*, CommentsCorrectionsList?, GeneSymbolList?, MeshHeadingList?, NumberOfReferences?, PersonalNameSubjectList?, OtherID*, OtherAbstract*, KeywordList*, SpaceFlightMission*, InvestigatorList?, GeneralNote*)⟩

Result of SOA2DescriptiveChare

⟨!ELEMENT MedlineCitation (PMID, DateCreated, DateCompleted, DateRevised?, Article, MedlineJournalInfo, ChemicalList?, SupplMeshList?, CitationSubset*, CommentsCorrectionsList?, GeneSymbolList?, MeshHeadingList?, NumberOfReferences?, PersonalNameSubjectList?, OtherID*, OtherAbstract*, KeywordList*, SpaceFlightMission*, InvestigatorList?, GeneralNote*)⟩

Example 3: MedlineCitation (Part 2/2)

MedlineCitation from the MEDLINE files

Official DTD

⟨!ELEMENT MedlineCitation (PMID, DateCreated, DateCompleted?, DateRevised?, Article, MedlineJournalInfo, ChemicalList?, SupplMeshList?, CitationSubset*, CommentsCorrectionsList?, GeneSymbolList?, MeshHeadingList?, NumberOfReferences?, PersonalNameSubjectList?, OtherID*, OtherAbstract*, KeywordList*, SpaceFlightMission*, InvestigatorList?, GeneralNote*)⟩

Result of SOA2DescriptiveSore

⟨!ELEMENT MedlineCitation (PMID, DateCreated, DateCompleted, DateRevised?, Article, MedlineJournalInfo, ChemicalList?, SupplMeshList?, CitationSubset*, CommentsCorrectionsList?, GeneSymbolList?, (MeshHeadingList, NumberOfReferences?)?, PersonalNameSubjectList?, (OtherID+, OtherAbstract*)?, KeywordList*, SpaceFlightMission*, InvestigatorList?, GeneralNote*)⟩

Summary

Summary

- our algorithms for learning CHAREs or SOREs...
  - generalize optimally (and less than previous algorithms)
  - are efficient (and more efficient than previous algorithms)
  - can be extended like the previous algorithms

Summary

Summary

- our algorithms for learning CHAREs or SOREs. . .
  - generalize optimally (and less than previous algorithms)
  - are efficient (and more efficient than previous algorithms)
  - can be extended like the previous algorithms
- we did not use results on descriptive generalization of pattern languages, but those results told us where to look

Possible extensions:

- numerical parameters
- integration into learning algorithms for other schema languages

Potential next steps:

- implementation and tests
- k-OREs
- learning regular expressions with backreferences (regex)

Final Slide

Thank you for your attention.