XTRACT: A System for Extracting Document Type Descriptors From XML Documents

Kyuseok Shim

KAIST

http://cs.kaist.ac.kr/~shim

(Joint work with Minos Garofalakis, Aristides Gionis, Rajeev Rastogi and S. Seshadri)

XML

- A W3C standard to complement HTML
- An instance of semistructured data [Abi97]
  - Document Type Descriptor (DTD)
- Origin: SGML
- Tags describe the semantics of the data
  - HTML simply specify how the data time is to be displayed
- An element can contain a sequence of nested sub-elements
- Sub-elements may themselves be tagged elements or character data

XML

- Standard for data representation and data exchange
- Looks like HTML but it isn’t
- Collection of elements
  - Atomic (raw character data)
  - Composite (sequence of nested sub-elements)
- Example

```
<book>
 <title> A Relational Model for Large Shared Data Banks </title>
 <author>
  <name E.F. Codd </name>
  <affiliation> IBM Research </affiliation>
 </author>
</book>
```

Document Type Definition

- A part of XML specification
- An XML document may have a DTD
- Grammar for describing the structure of XML document
- The structure of an element is specified by a regular expression
- Terminology for XML
  - well-formed: if tags are correctly closed
  - valid: if it has a DTD and conforms to it
- For exchanges of data, validation is useful

Document Type Definition

- Syntax
  - comma: sequence
  - |: or
  - (): grouping
  - ?, *, +: zero or one, zero or more, one or more occurrences
  - ANY: allows an arbitrary XML fragment to be nested within the element

An Example DTD

<!ELEMENT article (title, author*)>

<!ELEMENT title (#PCDATA)>

<!ELEMENT author (name, affiliation)>

<!ELEMENT name (#PCDATA)>

<!ELEMENT affiliation (#PCDATA)>

An Example XML Document

<article>

<title> A Relational Model for Large Shared Data Banks </title>

<author> E. F. Codd </name>

<affiliation> IBM Research </affiliation>

<author>

</article>

XTRACT System

- Goal:

  Infer DTDs from XML Documents

Why are DTDs useful?

- Plays a crucial role in efficient storage of XML data
  - [Deutsch, Fernandez, Suciu '99]
  - [Shanmugasundaram et al '99]
  - DTD is exploited to generate effective relational schema
- Optimization of XML queries
  - [Goldman, Widom '97]
  - [Fernandez, Suciu '97]
  - DTD allows to restrict the search only relevant portions of the data
- Aids users to form meaningful queries over the XML database

Why need to infer DTD?

- DTDs are not mandatory => might be missing
- XML databases that have been generated automatically
  – Relational databases
  – Flat files
  – HTML documents
  – Bibliographies

Related Work I

- Mining DTDs from a collection of XML documents has not been addressed in the literature
- Database Community
  - Extraction of schema from semistructured data
    - [Nestrorov, Abiteboul, Motwani ‘98]
    - [Goldman, Widom ’97]
    - [Fernandez, Suciu ’97]
- Attempts to find typing for semistructured data
- In DTD, outgoing edges from a type can be described by an arbitrary regular expression
- No ordering is imposed for edges

Related Work II

KAIST
한국과학기술원
Korea Advanced Institute of Science & Technology

- [Gold '67], [Gold '78], [Angluin '78], [Pitt '89]
  - Infer formal languages from examples
  - Purely theoretical and focus on investigating the computational complexity of the language inference problem
- [Kipelainen, Mannila, Ukkonen '95]
  - Infers a pattern language from positive examples
  - MDL principle was used
  - Assume the set of simple patterns is available
  - Cannot find general regular expressions
  - Patterns are not known apriori

Problem Simplification

- Element types => alphabet
- Infer DTD for each element type separately
- Example sequences: instances of nested subelements
  - Only one level down in the hierarchy

Problem Formulation

- Given a set example sequences for element *e*
- Compute a DTD for *e*

Hard problem since DTDs can be general, complex regular expressions

Example

```
<book>
  <title> </title>
  <author>
    <name> </name>
    <affiliation> </affiliation>
  </author>
</book>
```

```
<book>
  <title> </title>
  <author>
    <name> </name>
    <address> </address>
  </author>
  <author>
    <name> </name>
    <address> </address>
  </author>
  <editor> </editor>
</book>
```

```
<paper>
  <title> </title>
  <author>
    <name> </name>
    <affiliation> </affiliation>
  </author>
  <conference> </conference>
  <year> </year>
</paper>
```

Example (Ctd)

KAIST
한국과학기술원
Korea Advanced Institute of Science & Technology

Simplified Example Sequences

<book> :- <title><author>,
<title><author><author><editor>

<paper> :- <title><author><conference><year>

<author> :- <name><affiliation>,
<name><affiliation>,
<name><address>,
<name><address>

Desirable Solution

<!ELEMENT book (title, author*, editor?)
<!ELEMENT paper (title, author, conference, year)>
<!ELEMENT author (name, affiliation?, address?)>

Naive Approach I

- Factor as much as possible
  - e.g. t, ta, taa, taaa, taaaa
    - t | t (a| a(a | a(a | aa)))
    - much more voluminous and a lot less intuitive

Naive Approach II – DFA Minimization

- Suggestion:
  - Consider all example sequences as positive examples of a language of a DFA
  - Find the minimum DFA that accepts the language
- However:
  - Minimum DFA does not imply an intuitive simple regular expression
  - DFA minimization algorithm finds equivalent language
    - Does not generalize
  - How to infer ta* from ta, taa, taaa, taaaa?

Desirable DTD

- It generalize to intuitively correct but previously unseen examples
- It should be concise (i.e. small in size)
  - easy to understand and succinct
- It should be precise
  - not cover too many sequences not contained in the set of examples
  - not too general and captures the structure of input sequences

Trade-off!

Example

- p:- t, ta, taa, taaa, taaaa

Candidate DTD	Concise	Precise
ta|taa|taaa|taaaa	no	yes
(t|a)*	yes	No
ta*	yes	somewhat

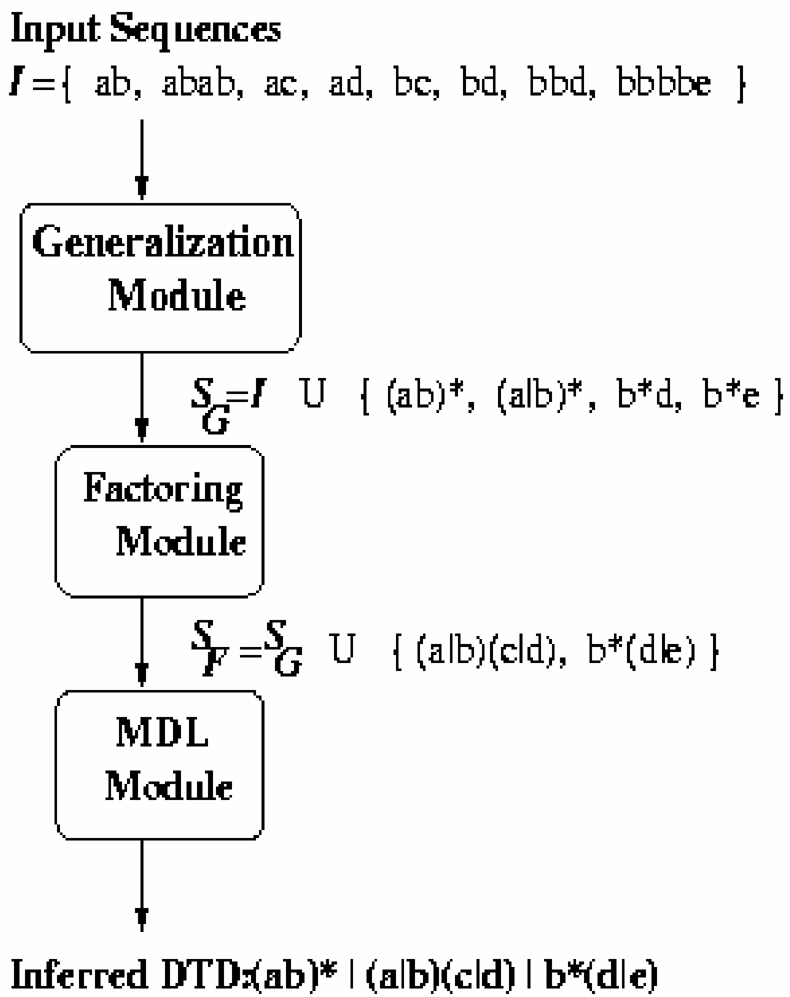
Example

I = {ab, abab, ababab}

- (a | b)*
  - a gross over-generalization of the input
  - completely fails to capture any structure inherent in input
- ab | abab | ababab, ab | ab(ab | abab)
  - accurately reflect the structure of the input sequences but do not generalize
- (ab)*
  - succinct and generalizes the input sequence without loosing too much structure information

Overview of XTRACT System

KAIST
한국과학기술원
Korea Advanced Institute of Science & Technology



XTRACT System

- Generalization
  - generalizes zero or more candidate DTDs by replacing patters in the input sequence with meta-characters like *
  - e.g. abab => (ab)*,   bbbe => b*e
- Factorization
  - factors common subexpressions from the generalized candidate DTDs
  - e.g. b*d | b*e => b* (d | e)
- Minimum Description Length (MDL) Principle
  - MDL ranks each candidate DTD and chooses the minimum cost DTD

MDL Principle

KAIST
한국과학기술원
Korea Advanced Institute of Science & Technology

- An information-theoretic measure for quantifying and thereby resolving the tradeoff between the conciseness and preciseness
- MDL principle has been successfully applied in a variety of situations
  - e.g. decision tree classifiers
- Roughly speaking, the best theory to infer from a set of data is the one that minimizes the sum of
  - the length of the theory, in bits (conciseness)
  - the length of the data, in bits, when encoded with the help of the theory (preciseness)

MDL Module

- MDL Principle: minimize the sum of
  - Theory description length, plus
  - Data description length given the theory

- In order to use MDL, need to:
  - Define theory description length (candidate DTD)
  - Define data description length (input sequences) given the theory (candidate DTD)
  - Solve the resulting minimization problem

MDL Module – Encoding Scheme

- Description Length of a DTD
  - Number of bits required to encode the DTD
  - |Size of DTD| x log (# of alphabets)
- Description Length of a sequence given a candidate DTD
  - Number of bits required to specify the sequence given DTD
  - Use a sequence of encoding indices
  - Encoding of *a* given *a* is the empty string
  - Encoding of *a* given (*a*|*b*|*c*) is the index 0
  - Encoding of *aaa* given a* is the index 3

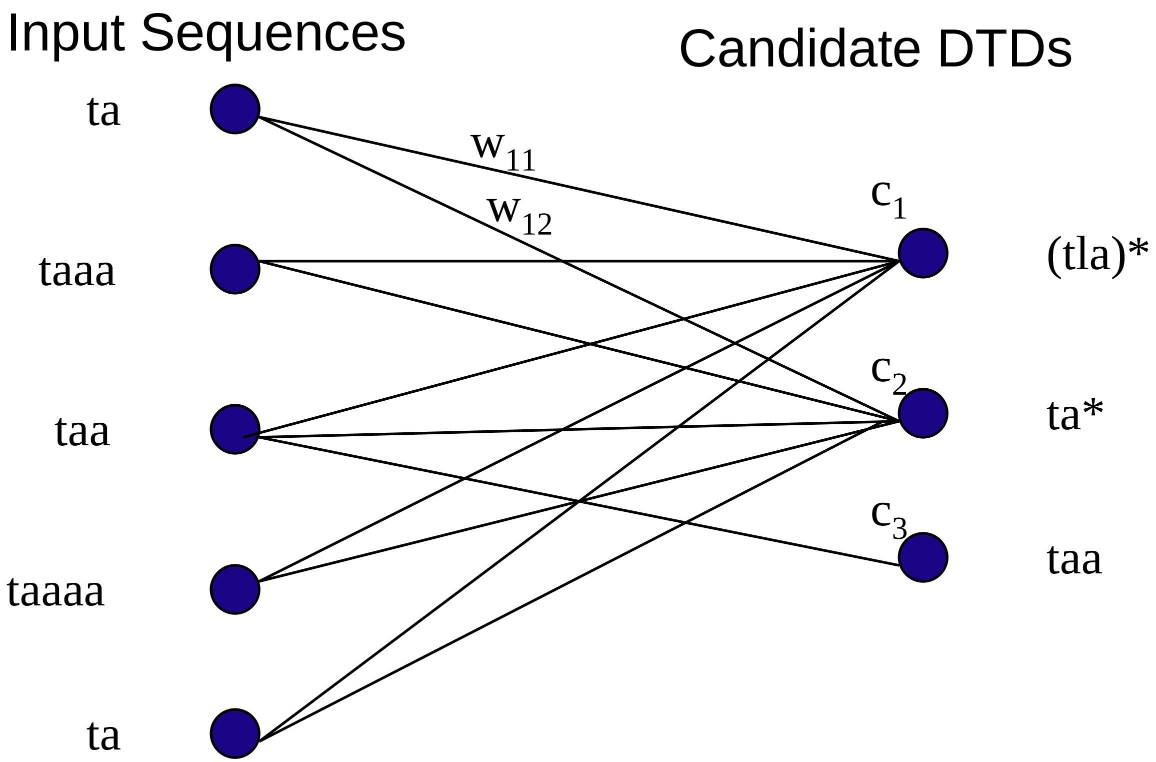
Example

I = {ab, abab, ababab}

- (a | b)*
  - abab: cost of 5 (the number of repetitions (4) + 4 characters to represent chosen character)
  - MDL cost = 6 (encoding DTD) + 3 + 5 + 7 = 21
- ab | abab | ababab
  - MDL cost = 14 + 3 = 17
- ab | ab(ab | abab)
  - MDL cost = 14 + 1 + 2 + 2 = 19
- (ab)*
  - MDL cost = 5 + 3 = 8

MDL Module - Minimization

Facility Location Problem (NP-hard)



Computing the DTD with Minimum MDL Cost

KAIST
한국과학기술원
Korea Advanced Institute of Science & Technology

- Facility Location Problem (FLP)
  - Let C be a set of clients and J be a set of facilities
  - c(j): cost of choosing a facility $j \in J$
  - d(j,i): cost of serving client $i \in C$ by facility $j \in J$
  - Choose a subset of facilities $F \subset J$ such that

$$\min_{F \subset J} \{ \sum_{j \in F} c(j) + \sum_{i \in C} \min_{j \in F} d(j,i) \}$$

  - NP Hard problem
  - Approximate algorithm [Charikar, Guha '99]: O(n^2logn)

Generalization Module

- Goal:
  - Introduce metacharacters *, |
  - Produce candidate DTDs of the form
  - a*bc*, (abc)*, (a|b|c)*, ((ab)*c)*, etc.
- Combine two generalization heuristics
  - DISCOVERSEQPATTERN(s,r)
    - S=*att…ttb* => *at*b*
  - DISCOVERORPATTERN(s,d)
    - Symbol a1, a2,…am in close proximity => (a1|a2|…|am)
- Candidate DTDs are generated by calling the above functions for appropriate values of r and d

Generalization - Example

- DISCOVERSEQPATTERN(s,r)

e.g. s = abababcabcababc, r = 2

abababcabcababc =>

(ab)*cabcababc =>

(ab)*cabc(ab)*c => ((ab)*c)*

- DISCOVERORPATTERN(s,d)

e.g. s = abcbac, d= 2

abcbac => a(b|c)*ac

e.g. s = abcbac, d = 3

abcbac => (a|b|c)*

Factoring Module

- Goal: Combine different candidates to derived more compact, factored DTDs forms
  - e.g. ac, ad, bc, bd => (a|b)(c|d)
- MDL modeling justification: Reduce description length of the candidate DTDs
- Intuitive justification: Capture DTDs like

  <!ELEMENT article (title, author*,

  (workshop | conference | journal),

  (computer science | physics | chemistry | …))>
- Capture optional elements
  - ab,a => a(b | ε) => ab?

Factoring Module

- Adaptation of the factoring algorithms for boolean expressions in logic optimization literature
- Novel heuristic algorithms for selecting subsets of candidate DTDs that give good factored forms

Experimental Result

- Tested Algorithms
  - XTRACT
  - DDbE: IBM alphaworks DTD extraction tool (V 1.0)
    - Java component library for inferring DTD from XML data instances
    - Control parameters: default values are used

- Use both Synthetic and Real-life DTDs

- Data Sets: Generate 1000 random example sequences for each DTD

Experimental Results

- Real-life Data Set
  - Newspaper Association of America (NAA) Classified Advertising Standard XML DTD

No.	Original DTD	Simplified DTD
1	< !ENTITY % included-elements "audio-clip|blind-box-reply|graphic|linkpi-char|video-clip" >	$a\|b\|c\|d\|e$
2	< !ELEMENT communications-contacts (phone|fax|email|pager|web-page)∗ >	$(a\|b\|c\|d\|e)^*$
3	< !ELEMENT employment-services(employment-service.type, employment-service.location ∗ (e.zz-generic-tag)∗) >	ab^*c^*
4	< !ENTITY % location"addr∗, geographic-area?, city?, state-province?, postal-code?, country?" >	$a^*b?c?d?$
5	< !ELEMENT transfer-info(transfer-number, (from-to, company-id)+, contact-info)∗ >	$(a(bc)^+d)^*$
6	< !ELEMENT real-estate-services(real-estate-service.type, real-estate-service.location?, r-e.response-modes∗, r-e.comment?)∗ >	$(ab^?c^*d?)^*$

Table 4: Real-life DTD Data Set

Experimental Result

KAIST 한국과학기술원 Korea Advanced Institute of Science & Technology

- Real-life Data Set
  - XTRACT is able to infer the first five correctly
  - DDbE could obtain the original DTD only the third one

NO	Simplified DTD	DTD Obtained by XTRACT	DTD obtained by DDbE
1	$a\|b\|c\|d\|e$	$a\|b\|c\|d\|e$	$a\|b\|c\|d\|e$
2	$(a\|b\|c\|d\|e)^*$	$(a\|b\|c\|d\|e)^*$	$(a\|b\|c\|d\|e)^*$
3	(ab^*c^*)	ab^*c^*	$(ab^+c^*)\|(ac^*)$
4	$a^*b?c?d?$	$a^*b?c?d?$	$(a^+b(c\|(c?d))?)\|((b\|a^+)?cd)\|$ $((a^+\|b)?d)\|((a^+\|b)?c)\|a^+\|b)$
5	$(a(bc)^+d)*$	$(a(bc)^*d)^*$	$(a\|b\|c\|d)^+$
6	$(ab?c^*d?)*$	–	$(a\|b\|c\|d)+$

Table 6: DTDs generated by XTRACT and DDbE for Real-life Data Set

Experimental Results

- Synthetic Data Set
  - Each DTD is randomly chosen from either
    - $A_1|A_2|\ldots|A_n$ or $A_1A_2A_3\ldots A_n$
    - n is randomly chosen between a1 and mb
    - Each A_i has one of the following four forms
      - $(a_1|a_2|\ldots|a_{mi})$
      - $(a_1a_2\ldots A_{mi})$
      - $(a_1|a_2|\ldots|a_{mi})^*$
      - $(a_1a_2\ldots a_{mi})^*$
    - m_i is chosen to be between 1 and ms

Experimental Results

- Synthetic Data Set
  - abcde | efgh | ij | klm
  - (a | b | c | d | f)*gh
  - (a | b | c | d)* | e
  - (abcde)*f
  - (ab)* | cdef | (ghi)*
  - abcdef(g | h | I | j)(k | l | m | n | o)
  - (a | b | c)d*e*(fgh)*
  - (a|b)(cdefg)*hijklmnopq(r|s)*
  - (abcd)*|(e|f|g)*| h | (I j k l m)*
  - a* | (b | c | d | e | f)* | gh | (I | j | k)* | (lmn)

Experimental Result

Synthetic Data Set

No.	Original DTD	DTD Inferred by XTRACT	DTD Inferred by DDbE
1	$abcde \mid efgh \mid ij \mid klm$	$abcde \mid efgh \mid ij \mid klm$	$abcde \mid efgh \mid ij \mid klm$
2	$(a \mid b \mid c \mid d \mid f)^* gh$	$(a \mid b \mid c \mid d \mid f)^* gh$	$gh(a \mid b \mid c \mid d \mid f)^+ gh$
3	$(a \mid b \mid c \mid d)^* \mid e$	$(a \mid b \mid c \mid d)^* \mid e$	$(e(a \mid c \mid d \mid b)^+ e)$
4	$(abcde)^* f$	$(abcde)^* f$	$(f(a \mid e \mid d \mid c \mid b)^+ f)$
5	$(ab)^* \mid cdef \mid (ghi)^*$	$(ab)^* \mid cdef \mid (ghi)^*$	$cdef(a \mid b \mid g \mid i \mid h)^+ cdef$
6	$abcdef(g \mid h \mid i \mid j)(k \mid l \mid m \mid n \mid o)$	$abcdef(g \mid h \mid i \mid j)(k \mid l \mid m \mid n \mid o)$	$abcdef(j(o \mid l \mid m \mid n \mid k) \mid g(o \mid l \mid n \mid m \mid k) \mid$ $h(m \mid l \mid n \mid k \mid o) \mid i(o \mid l \mid n \mid m \mid k))$
7	$(a \mid b \mid c)d^* e^* (fgh)^*$	$(a \mid b \mid c)d^* e^* (fgh)^*$	$((c \mid b \mid a)d^+ e^+ \mid ad^+ \mid bd^+ \mid c(e^+ \mid d^+)? \mid$ $ad^* \mid be^*))(f \mid h \mid g)^+ ((a \mid b \mid c)d^+ e^+ \mid$ $c(e^+ \mid d^+)? \mid a(e^+ \mid d^+)? \mid b(e^+ \mid d^+)?)$
8	$(a \mid b)(cdefg)^*$ $hijklmnopq(r \mid s)^*$	$(a \mid b)(cdefg)^*$ $hijklmnopq(r \mid s)^*$	$((((a \mid b)hijabcdefg) \mid b \mid a)$ $(c \mid g \mid f \mid e \mid d \mid s \mid r)^+ ((b \mid a)?hijkamnopq))$
9	$(abcd)^* \mid (e \mid f \mid g)^* \mid h \mid (ijklm)^*$	$(abcd)^* \mid (ijklm)^* \mid h \mid (e \mid f \mid g)^*$	$h(a \mid d \mid c \mid b \mid e \mid g \mid f \mid i \mid m \mid l \mid k \mid j)^+ h$
10	$a^* \mid (b \mid c \mid d \mid e \mid f)^* \mid gh \mid (i \mid j \mid k)^* \mid$ $(lmn)^*$	$a^* \mid (b \mid c \mid d \mid e \mid f)^* \mid gh \mid (i \mid j \mid k)^* \mid$ $(lmn)^*$	$(a^+ \mid gh)(e \mid f \mid d \mid i \mid j \mid l \mid n \mid m \mid k \mid c \mid b)^+$ $(a^+ \mid gh)$

Table 5: DTDs generated by XTRACT and DDbE for Synthetic Data Set

Experimental Result

- Synthetic Data Set
  - XTRACT finds all of original DTDs
  - DDbe discovers only the first DTD accurately
  - DDbe may not cover every input sequences
    - e.g. *gh* is not covered for Dataset 2
  - Note: Dataset 4
    - *f* is appended to both the front and the back
    - symbols repeated frequently are all *or*'d together
  - Ddbe is not very good at factoring
    - e.g. Dataset 6

Conclusions

- Inference of DTDs can be used for improving database storage and for query optimization
- Hard problem! – Naive approaches fail to produce intuitive, meaningful DTDs
- Presented the architecture of the XTRACT system for inferring a DTD for a database of XML documents
  – generalization step
  – factorization step
  – MDL cost selection step
- Demonstrated the effectiveness of XTRACT