
A Hierarchical Approach to Model Web Query Interfaces for Web Source Integration

Eduard Constantin Dragut*, Thomas Kabisch+, Clement Yu*, Ulf Leser+

* University of Illinois at Chicago

+ Humboldt-Universität zu Berlin

35th Intl. Conference on Very Large Data Bases

Lyon, France, August 2009

Outline

1. **Motivation**

- Integration of Deep Web Sources
- Hierarchical Schema Model

2. Concepts

3. Observations on Query Interfaces

4. Extraction Algorithm

5. Evaluation

6. Conclusion and Outlook



Motivation (I): Deep Web Integration

Search for Discount Airline Tickets

Depart From: Going To: Depart date: Feb 12 Time: Noon
Return from: Return To: Return date: Feb 19 Time: Noon

Roundtrip: Oneway: Economy

ADULT [Age 12 & ABOVE] **CHILD** [AGE 2 -11] **INFANT** [1 AND BELOW]
1 0 0

Airline: All NonStop

[Search Multiple Destinations](#)
[Business Class \(International only\)](#)

Online Reservations

From:
To:

One way Round trip

Trip Type [Multiple Destinations](#)

Depart: February 2009 9
Return: February 2009 11
Cabin: Coach Adult(s): 1

powered by abstravel

[More Options](#)

Round-trip Reservations

[One-way & multi-city reservations](#)

Leaving from: Departure date: Feb 26
Going to: Return date: Mar 05

Passengers: 1 Preferred cabin: Economy/Coach

Pricing - leave unchecked to find lowest fares
 [Unrestricted](#)

Round Trip One Way Multi-City

* Departing From: * Going To:

Leave On: THU 02/12 2009 Return On: SUN 02/15 2009

Adt: (12+) 1 Chd: (2-11) 0 Cabin Preference: Economy

Book return flights

[Book one-way and multi-city](#) ▶
[Tips for finding low fares](#) ▶

Leaving From (city or airport)

Going To (city or airport)

Adults (12+ yrs) 1 Children (2-11 yrs) 0 Infants (<2 yrs) 0

Departing On 13 Mar
Returning On 20 Mar

[Flight class](#) ▶
Economy (lowest price)

[Get flights](#) ▶

Motivation (II): Deep Web Integration Process

1. **Extraction of Source Interface Schemata**
2. Derivation of integrated Interface Schema
3. Formulation of Queries
4. Transformation of Queries
5. Integration of Results

Motivation (III): Extraction of Interface Schema Trees

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Feb | 19 | Morning

Return Date

Feb | 26 | Morning

3. Number of Passengers:

Maximum 6 passengers per reservation.

Adults Children (ages 2-11)

1 0

4. What Are Your Service Preferences?

Class of Service:

Economy with Restrictions
 Economy without Restrictions
 Full Fare Economy Class
 Business Class
 First Class +

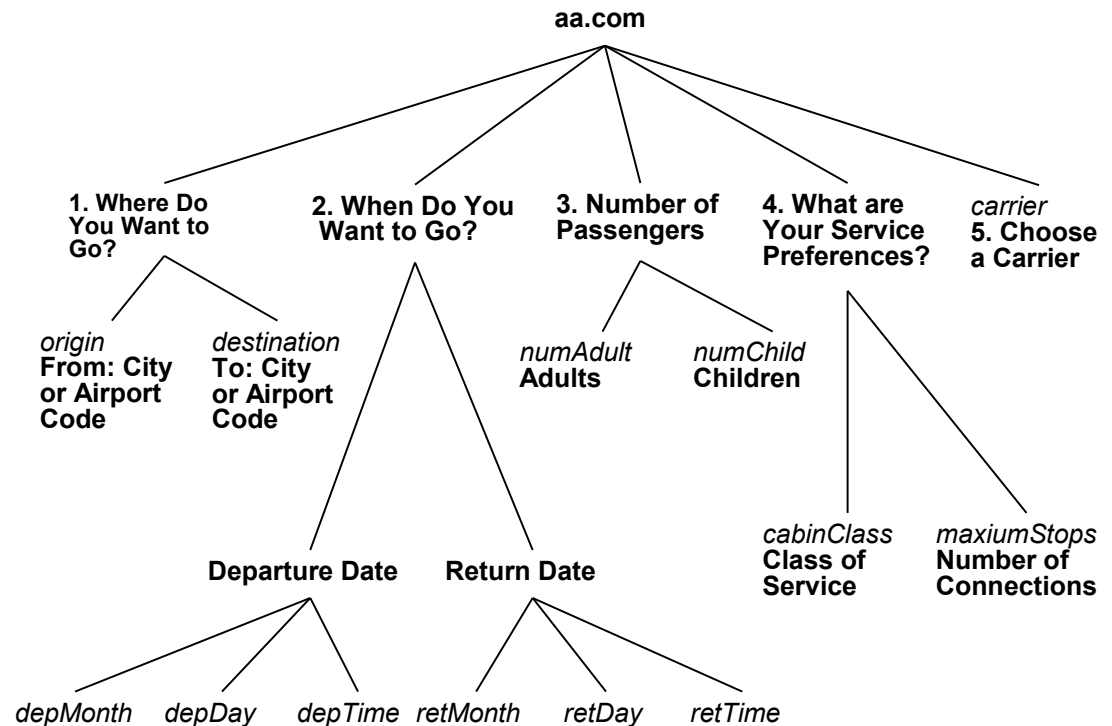
Number of Connections:

No Preference
 None
 1 or Less

+ For premium class travel to/from Canada, Caribbean, Mexico, Central America and Northern South America, please select Business Class.

5. Choose a Carrier:

All AA, Eagle, and AmericanConnection?
 All Carriers



Motivation (IV): Hierarchical Representation

Prior Work [BEP04, WISE05]

- flat representation
- no order
- no context

Improvements with ordered Schema Trees

Query Interface Matching

- handling of unlabelled fields
- resolving of homonymy problem
- identifying complex (1:n) field-mappings

Deep Web Crawling

- enrich annotations to fields
- identify interrelated fields

Outline

1. Motivation

2. Concepts

- Tokens
- Fields
- Labels

3. Observations on Query Interfaces

4. Extraction Algorithm

5. Evaluation

6. Conclusion and Outlook

Concepts (I): Tokens and Bounding Boxes

Token

- Def: visible piece of HTML code with layout properties
- Attributes: style Information, bounding box, content
- Types: text token, field token, image token

Bounding Box

- rectangular area that describes position and size of token
- pixel coordinates given by browser rendering engine

2. When Do You Want to Go?

Departure Date

Feb	▼	19	▼	Morning	▼
-----	---	----	---	---------	---

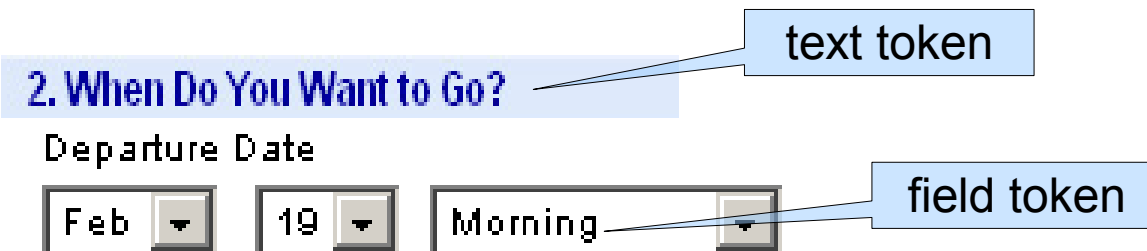
Concepts (I): Tokens and Bounding Boxes

Token

- Def: visible piece of HTML code with layout properties
- Attributes: style Information, bounding box, content
- Types: text token, field token, image token

Bounding Box

- rectangular area that describes position and size of token
- pixel coordinates given by browser rendering engine



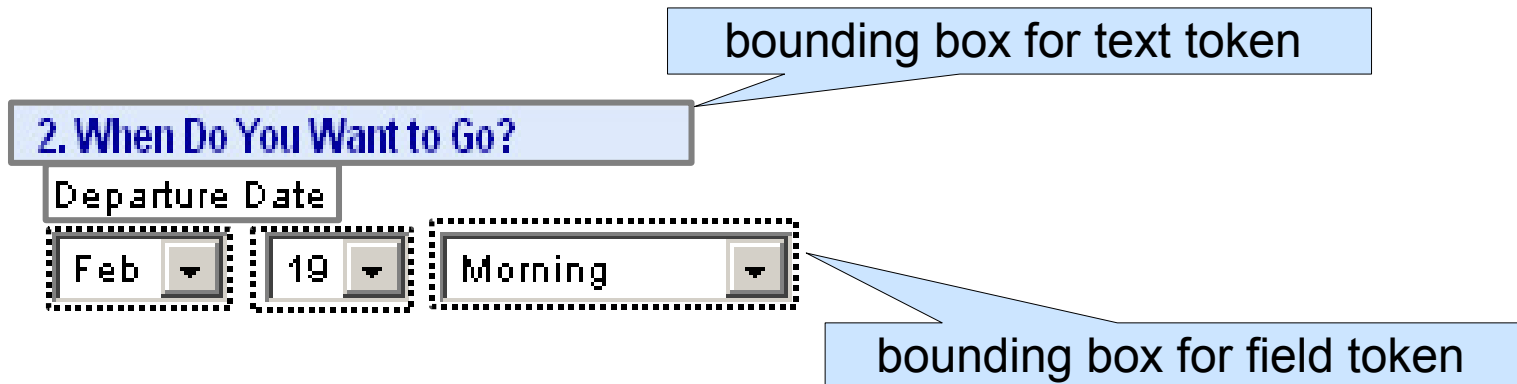
Concepts (I): Tokens and Bounding Boxes

Token

- Def: visible piece of HTML code with layout properties
- Attributes: style Information, bounding box, content
- Types: text token, field token, image token

Bounding Box

- rectangular area that describes position and size of token
- pixel coordinates given by browser rendering engine



Concepts (II): Fields and Labels

Field

- Def: interactive HTML form elements
- Important Types: input field, combo box, radio button...
- Attributes: name, values, default values

From: City or Airport Code To: City or Airport Code

Departure Date

Feb 19 Morning

Class of Service:

- Economy with Restrictions
- Economy without Restrictions
- Full Fare Economy Class
- Business Class
- First Class +

Label

- Def: texts denoting fields or groups of fields
- Types: field label, group label
- Properties: style, orientation

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

Concepts (II): Fields and Labels

Field

- Def: interactive HTML form elements
- Important Types: input field, combo box, radio button...
- Attributes: name, values, default values

From: City or Airport Code To: City or Airport Code

Departure Date

Feb 19 Morning

Class of Service:

- Economy with Restrictions
- Economy without Restrictions
- Full Fare Economy Class
- Business Class
- First Class +

A blue callout box labeled "input field" points to the empty text input field under "From: City or Airport Code".

Label

- Def: texts denoting fields or groups of fields
- Types: field label, group label
- Properties: style, orientation

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

Two empty text input fields are shown below the labels.

Concepts (II): Fields and Labels

Field

- Def: interactive HTML form elements
- Important Types: input field, combo box, radio button...
- Attributes: name, values, default values

The screenshot shows a flight booking interface. At the top, there are two input fields labeled "From: City or Airport Code" and "To: City or Airport Code". Below these is a "Departure Date" section with three dropdown menus: "Feb", "19", and "Morning". A blue callout box labeled "combo box" points to the "Morning" dropdown. Below the date section is a "Class of Service:" section with four radio button options: "Economy with Restrictions" (selected), "Economy without Restrictions", "Full Fare Economy Class", "Business Class", and "First Class +".

Label

- Def: texts denoting fields or groups of fields
- Types: field label, group label
- Properties: style, orientation

The screenshot shows a form section titled "1. Where Do You Want to Go?". It contains two input fields, one labeled "From: City or Airport Code" and one labeled "To: City or Airport Code".

Concepts (II): Fields and Labels

Field

- Def: interactive HTML form elements
- Important Types: input field, combo box, radio button...
- Attributes: name, values, default values

From: City or Airport Code To: City or Airport Code

Departure Date

Feb 19 Morning

Class of Service:

- Economy with Restrictions
- Economy without Restrictions
- Full Fare Economy Class
- Business Class
- First Class +

radio button

Label

- Def: texts denoting fields or groups of fields
- Types: field label, group label
- Properties: style, orientation

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

Concepts (II): Fields and Labels

Field

- Def: interactive HTML form elements
- Important Types: input field, combo box, radio button...
- Attributes: name, values, default values

From: City or Airport Code To: City or Airport Code

Departure Date

Feb 19 Morning

Class of Service:

- Economy with Restrictions
- Economy without Restrictions
- Full Fare Economy Class
- Business Class
- First Class +

Label

- Def: texts denoting fields or groups of fields
- Types: field label, group label
- Properties: style, orientation

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

field label

Concepts (II): Fields and Labels

Field

- Def: interactive HTML form elements
- Important Types: input field, combo box, radio button...
- Attributes: name, values, default values

From: City or Airport Code To: City or Airport Code

Departure Date

Feb 19 Morning

Class of Service:

- Economy with Restrictions
- Economy without Restrictions
- Full Fare Economy Class
- Business Class
- First Class +

Label

- Def: texts denoting fields or groups of fields
- Types: field label, group label
- Properties: style, orientation

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

group label

Concepts (II): Fields and Labels

Field

- Def: interactive HTML form elements
- Important Types: input field, combo box, radio button...
- Attributes: name, values, default values

From: City or Airport Code To: City or Airport Code

Departure Date

Feb 19 Morning

Class of Service:

- Economy with Restrictions
- Economy without Restrictions
- Full Fare Economy Class
- Business Class
- First Class +

Label

- Def: texts denoting fields or groups of fields
- Types: field label, group label
- Properties: style, orientation

label style #1 (e.g. blue, 14 px)

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

label style #2 (e.g. black, 10 px)

Concepts (II): Fields and Labels

Field

- Def: interactive HTML form elements
- Important Types: input field, combo box, radio button...
- Attributes: name, values, default values

From: City or Airport Code To: City or Airport Code

Departure Date

Feb 19 Morning

Class of Service:

- Economy with Restrictions
- Economy without Restrictions
- Full Fare Economy Class
- Business Class
- First Class +

Label

- Def: texts denoting fields or groups of fields
- Types: field label, group label
- Properties: style, orientation

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

label oriented above field

Outline

1. Motivation

2. Concepts

3. **Observations on Query Interfaces**

- **Order and Alignment**
- **Field Scope**
- **Label Scope**

4. Extraction Algorithm

5. Evaluation

6. Conclusion and Outlook

Observations (I): Order and Alignment of Fields

Order

- defined by their fill-in order
- given by designer (tab-order, code order)

Alignment

- hint for semantic relation
- horizontal / vertical
- between consecutive fields only

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Observations (I): Order and Alignment of Fields

Order

- defined by their fill-in order
- given by designer (tab-order, code order)

Alignment

- hint for semantic relation
- horizontal / vertical
- between consecutive fields only

The image shows a screenshot of a web form with two sections. The first section is titled "1. Where Do You Want to Go?" and contains two input fields: "From: City or Airport Code" and "To: City or Airport Code". The second section is titled "2. When Do You Want to Go?" and contains a "Departure Date" field with three sub-fields: "Feb", "10", and "morning". A blue box labeled "field order" has an arrow pointing to the "morning" field. Another arrow points from the "From" field to the "To" field, and a third arrow points from the "From" field to the "Feb" field.

Observations (I): Order and Alignment of Fields

Order

- defined by their fill-in order
- given by designer (tab-order, code order)

Alignment

- hint for semantic relation
- horizontal / vertical
- between consecutive fields only

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Feb 19 Morning

horizontal alignment

Observations (I): Order and Alignment of Fields

Order

- defined by their fill-in order
- given by designer (tab-order, code order)

Alignment

- hint for semantic relation
- horizontal / vertical
- between consecutive fields only

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Feb 19 Morning

no vertical alignment

Observations (II): Field Scope

Field Scope

- area where a field candidate label may be placed
- contains four scope sectors

Scope Sector

- ranges from field to the left, to the right, upwards or downwards
- bound by other fields or interface boundary

Return Date

Feb 28 Morning

3. Number of Passengers:

Maximum 6 passengers per reservation.

Adults Children (ages 2-11)

1 0

4. What Are Your Service Preferences?

Class of Service:

Economy with Restrictions

Economy without Restrictions

Number of

No P

Nonx

Observations (II): Field Scope

Field Scope

- area where a field candidate label may be placed
- contains four scope sectors

Scope Sector

- ranges from field to the left, to the right, upwards or downwards
- bound by other fields or interface boundary

The image shows a screenshot of a flight booking form with several fields and sections. A dashed box highlights a specific area containing the following elements:

- Return Date:** Feb, 28, Morning
- 3. Number of Passengers:** A section with a title and sub-headers "Maximum # passengers per reservation." and "Adults Children (ages 2-11)".
- 1 0:** Two dropdown menus for the number of passengers.
- 4. What are your Service Preferences?:** A section with a title and sub-headers "Class of Service:" and "Number of".
- Options:** Radio buttons for "Economy with Restrictions", "Economy without Restrictions", "No P", and "None".

Annotations on the screenshot identify four scope sectors:

- C1:** A vertical dashed box on the left side of the "3. Number of Passengers" section.
- ST:** A vertical solid box between "3. Number of Passengers" and "4. What are your Service Preferences?".
- C2:** A horizontal dashed box on the right side of the "3. Number of Passengers" section.
- SL:** A horizontal solid box on the left side of the "1 0" dropdowns.
- SR:** A horizontal solid box on the right side of the "1 0" dropdowns.
- C3:** A vertical dashed box on the left side of the "4. What are your Service Preferences?" section.
- SB:** A vertical solid box between "4. What are your Service Preferences?" and the "1 0" dropdowns.
- C4:** A horizontal dashed box on the right side of the "4. What are your Service Preferences?" section.

Callout boxes point to the "top scope sector" (C2) and the "blind scope sector" (SR).

Observations (III): Label Scope

Intuition

- query interfaces are organized top-down and left-right
- labels have area where they are visible

Definition

- rectangular area ranging from the (top,left)-corner of the label bounding box to the right and to the bottom until the next label of the same text style or the interface boundary
- never intersect other label's scope

2. When Do You Want to Go?

Departure Date

Feb 19 Morning

Return Date

Feb 26 Morning

3. Number of Passengers:

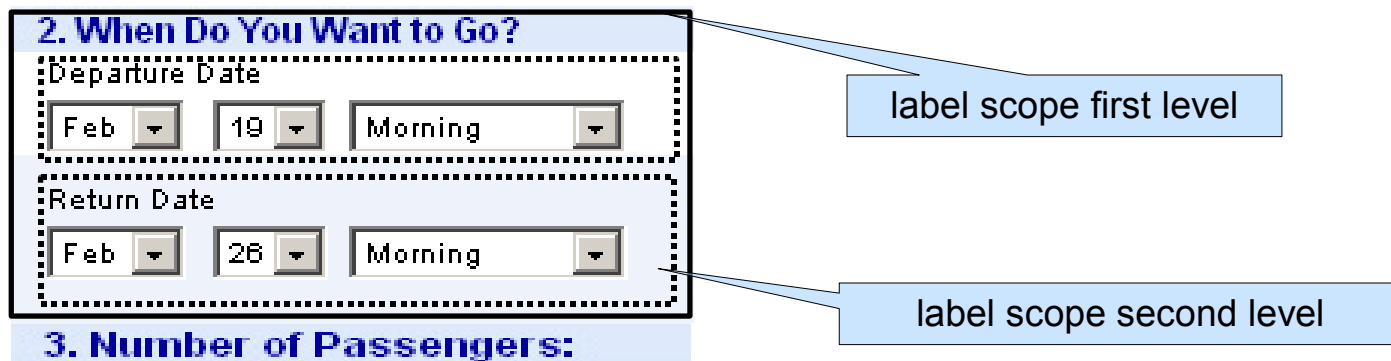
Observations (III): Label Scope

Intuition

- query interfaces are organized top-down and left-right
- labels have area where they are visible

Definition

- rectangular area ranging from the (top,left)-corner of the label bounding box to the right and to the bottom until the next label of the same text style or the interface boundary
- never intersect other label's scope



Outline

1. Motivation
2. Concepts
3. Observations on Query Interfaces
4. **Extraction Algorithm**
 - **Field Tree Construction**
 - **Label Tree Construction**
 - **Tree Integration**
5. Evaluation
6. Conclusion and Outlook



Algorithm (I): Tree of Fields Construction

1. **add for each field a node into the empty tree FT**
2. estimate inflection points of field order
3. group all fields between two consecutive inflection points
4. insert an internal node N for each group of fields into tree of fields FT such that N becomes the direct parent node of all field nodes in the group

1. Where Do You Want to Go?

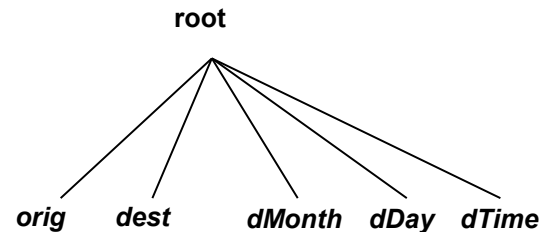
From: City or Airport Code To: City or Airport Code

→

2. When Do You Want to Go?

Departure Date:

Feb 10 morning



Algorithm (I): Tree of Fields Construction

1. add for each field a node into the empty tree FT
2. **estimate inflection points of field order**
3. group all fields between two consecutive inflection points
4. insert an internal node N for each group of fields into tree of fields FT such that N becomes the direct parent node of all field nodes in the group

1. Where Do You Want to Go?

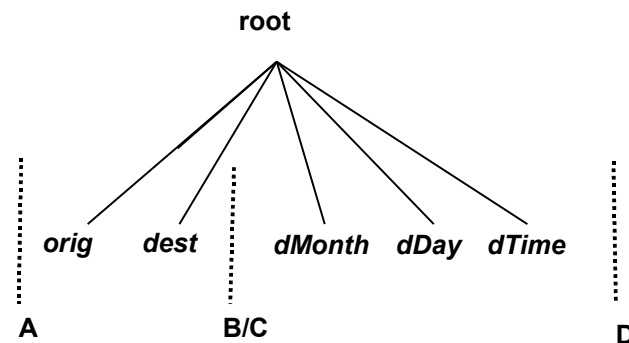
From: City or Airport Code To: City or Airport Code

A **B**

2. When Do You Want to Go?

Departure Date:

C **D**



Algorithm (I): Tree of Fields Construction

1. add for each field a node into the empty tree FT
2. estimate inflection points of field order
3. **group all fields between two consecutive inflection points and insert an internal node N for each group into FT such that N becomes parent node of all field nodes in the group**

1. Where Do You Want to Go?

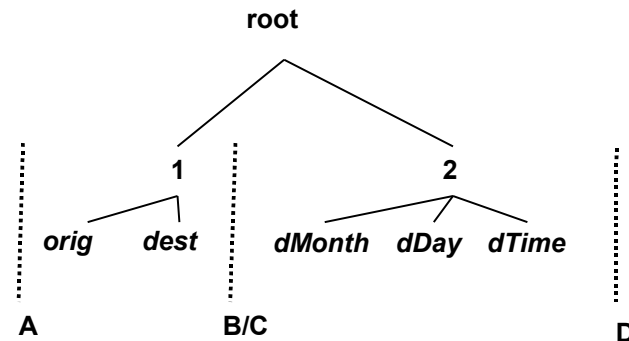
From: City or Airport Code To: City or Airport Code

A **B**

2. When Do You Want to Go?

Departure Date:

C **D**



Algorithm (II): Tree of Labels Construction

- 1.initialize label scope
- 2.get top cluster: smallest cluster of labels that covers all fields of current window
- 3.insert node for each element of top cluster as children of current root
- 4.recursively repeat for next lower cluster of labels

1. Where Do You Want to Go?

From: City or Airport Code

To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Feb ▾ 19 ▾ Morning ▾

Algorithm (II): Tree of Labels Construction

1. initialize label scope

2. get top cluster: smallest cluster of labels that covers all fields of current window
3. insert node for each element of top cluster as children of current root
4. recursively repeat for next lower cluster of labels

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Feb ▼ 19 ▼ Morning ▼

Algorithm (II): Tree of Labels Construction

1. initialize label scope

2. get top cluster: smallest cluster of labels that covers all fields of current window
3. insert node for each element of top cluster as children of current root
4. recursively repeat for next lower cluster of labels

1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Feb 19 Morning

Algorithm (II): Tree of Labels Construction

- 1.initialize label scope
- 2.**get top cluster: smallest cluster of labels that covers all fields of current window**
- 3.insert node for each element of top cluster as children of current root
- 4.recursively repeat for next lower cluster of labels

The image shows a screenshot of a flight search interface. It is divided into two main sections, each with a blue header and a white body.

1. Where Do You Want to Go?
This section contains two input fields. The first is labeled "From: City or Airport Code" and the second is labeled "To: City or Airport Code". Both fields are currently empty.

2. When Do You Want to Go?
This section is titled "Departure Date" and contains three dropdown menus. The first dropdown is set to "Feb", the second to "19", and the third to "Morning".

Algorithm (II): Tree of Labels Construction

- 1.initialize label scope
- 2.estimate top cluster: smallest cluster of labels that covers all fields of current window
- 3.insert node for each element of top cluster as children of current root**
- 4.recursively repeat for next lower cluster of labels

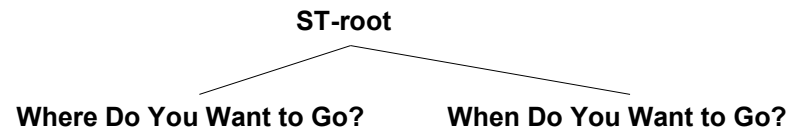
1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Feb ▾ 19 ▾ Morning ▾



Algorithm (II): Tree of Labels Construction

- 1.initialize label scope
- 2.estimate top cluster: smallest cluster of labels that covers all fields of current window
- 3.insert node for each element of top cluster as children of current root
- 4.**recursively repeat for next lower cluster of labels**

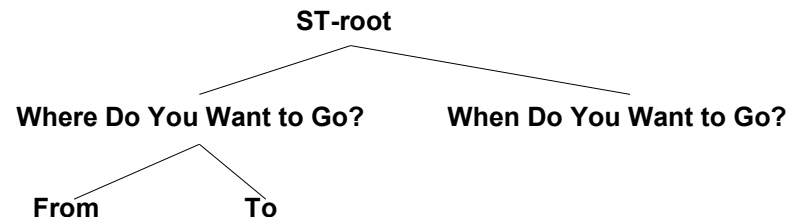
1. Where Do You Want to Go?

From: City or Airport Code To: City or Airport Code

2. When Do You Want to Go?

Departure Date

Feb 19 Morning



Algorithm (III): Integration of Trees

1. calculate for a label l the list of covered fields Df based on the label scope of l
2. identify a node lca in tree of fields FT such that lca is the lowest common ancestor of all the fields in Df and lca does not have any other descending leaves
3. map label to lca , distinguish three cases:
 - if a node lca can be identified in FT map l to lca , l becomes label of lca
 - if there is no such node and no group conflict insert lca into FT
 - if there is a group conflict resolve conflict and insert lca into FT

2. When Do You Want to Go?

Departure Date

Return Date

3. Number of Passengers:

Algorithm (III): Integration of Trees

1. **calculate for a label l the list of covered fields Df based on the label scope of l**
2. identify a node lca in tree of fields FT such that lca is the lowest common ancestor of all the fields in Df and lca does not have any other descending leaves
3. map label to lca , distinguish three cases:
 - if a node lca can be identified in FT map l to lca , l becomes label of lca
 - if there is no such node and no group conflict insert lca into FT
 - if there is a group conflict resolve conflict and insert lca into FT

2. When Do You Want to Go?

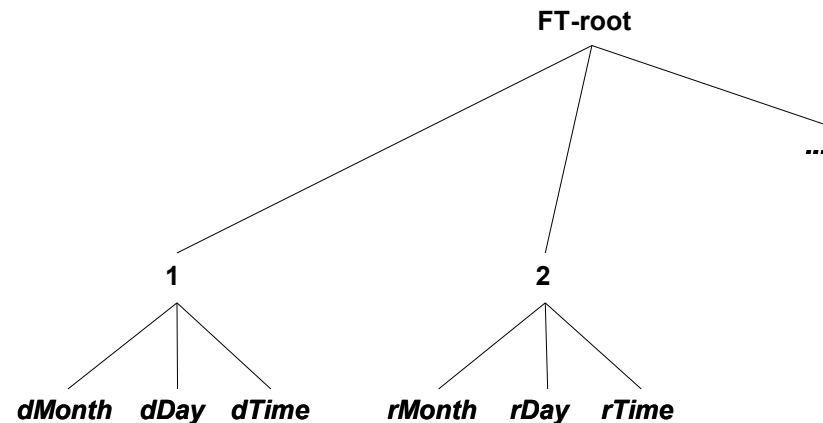
Departure Date

Feb 19 Morning

Return Date

Feb 26 Morning

3. Number of Passengers:



Algorithm (III): Integration of Trees

1. calculate for a label l the list of covered fields Df based on the label scope of l
- 2. identify a node lca in tree of fields FT such that lca is the lowest common ancestor of all the fields in Df and lca does not have any other descending leaves**
3. map label to lca , distinguish three cases:
 - if a node lca can be identified in FT map l to lca , l becomes label of lca
 - if there is no such node and no group conflict insert lca into FT
 - if there is a group conflict resolve conflict and insert lca into FT

2. When Do You Want to Go?

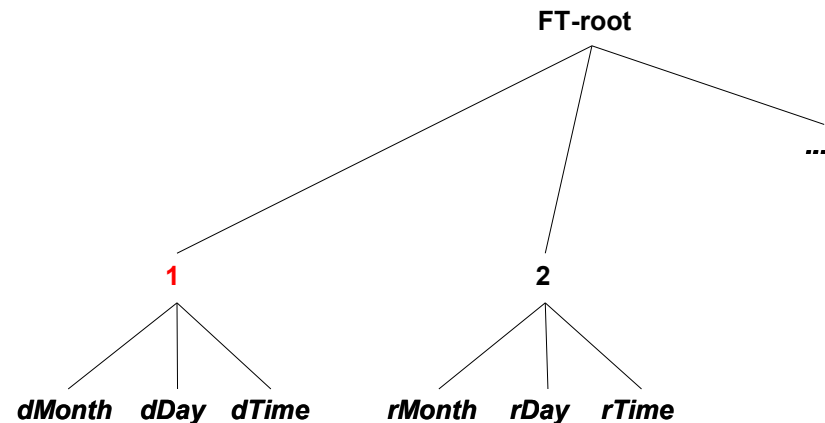
Departure Date

Feb 19 Morning

Return Date

Feb 26 Morning

3. Number of Passengers:



Algorithm (III): Integration of Trees

1. calculate for a label l the list of covered fields Df based on the label scope of l
2. identify a node lca in tree of fields FT such that lca is the lowest common ancestor of all the fields in Df and lca does not have any other descending leaves
3. **map label to lca , distinguish three cases:**
 - **if a node lca can be identified in FT map l to lca , l becomes label of lca**
 - if there is no such node and no group conflict insert lca into FT
 - if there is a group conflict resolve conflict and insert lca into FT

2. When Do You Want to Go?

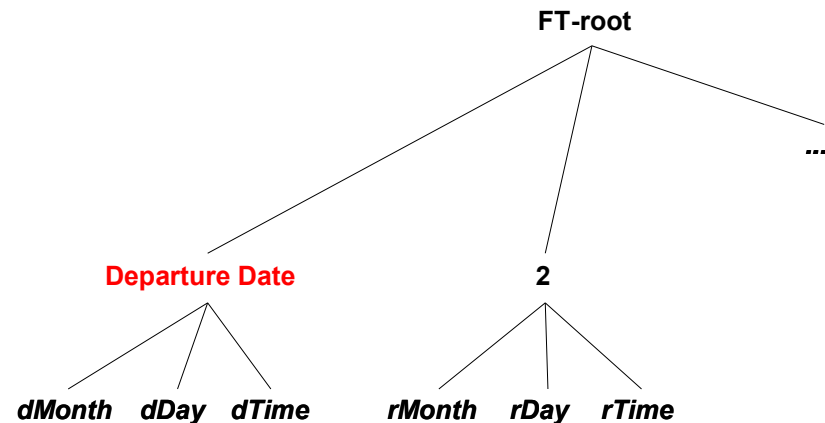
Departure Date

Feb 19 Morning

Return Date

Feb 26 Morning

3. Number of Passengers:



Algorithm (III): Integration of Trees

1. calculate for a label l the list of covered fields Df based on the label scope of l
2. identify a node lca in tree of fields FT such that lca is the lowest common ancestor of all the fields in Df and lca does not have any other descending leaves

3. map label to lca , distinguish three cases:

- if a node lca can be identified in FT map l to lca , l becomes label of lca
- **if there is no such node and no group conflict insert lca into FT**
- if there is a group conflict resolve conflict and insert lca into FT

2. When Do You Want to Go?

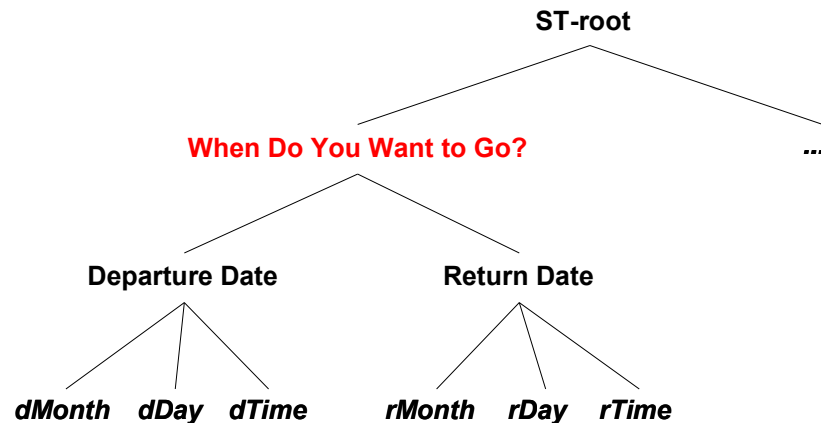
Departure Date

Feb 19 Morning

Return Date

Feb 26 Morning

3. Number of Passengers:



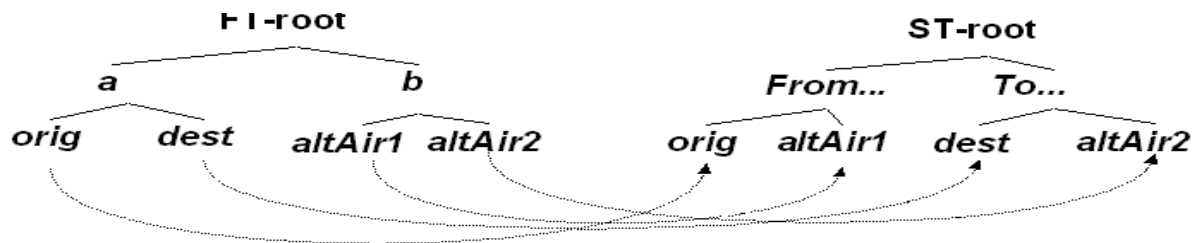
Algorithm (III): Integration / Conflict Resolution

- If field tree and label tree grouping conflict, the label tree grouping is given priority
- grouping in field tree may be design error (see example)

1. Where Do You Want to Go?

From: [City](#) or [Airport Code](#) To: [City](#) or [Airport Code](#)

and airports within and airports within



Field Tree

Schema Tree

Outline

1. Motivation

2. Concepts

3. Observations on Query Interfaces

4. Extraction Algorithm

5. **Evaluation**

- **Data Sets**
- **Results**

6. Conclusion and Outlook



Evaluation (I): Datasets

ICQ

- totals 100 query interfaces of five domains (e.g. airlines, real estate)
- each domain contains 20 interfaces
- collected to evaluate matching techniques [CLUST04]
- provides gold standard trees
- more complex, avg. number of internal nodes in gold standard 2.35

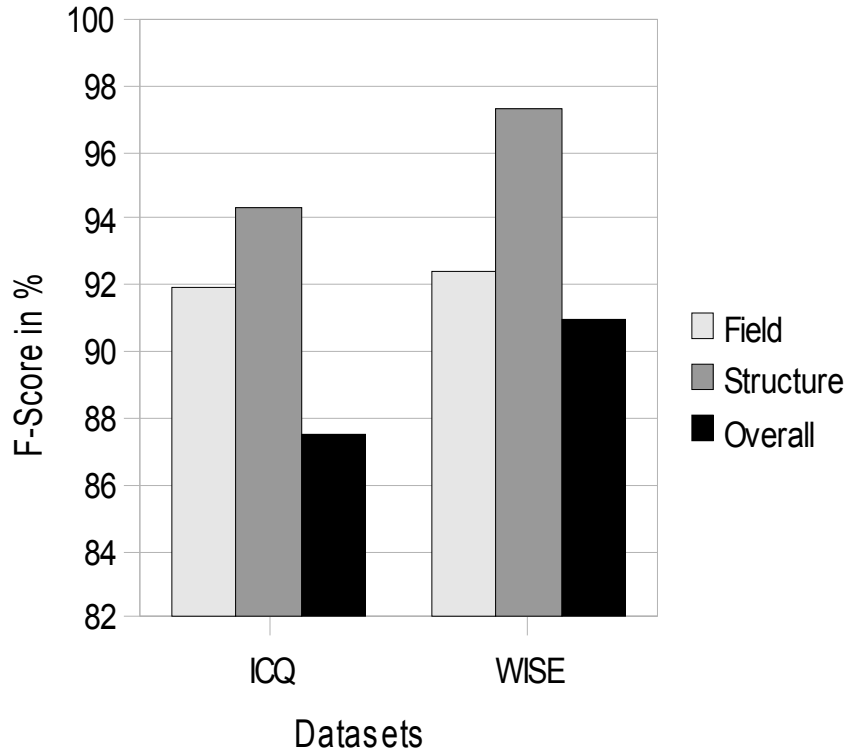
WISE

- 147 (135 accessible) interfaces of seven domains (e.g. books, watches)
- collected to evaluate WISE Extractor [WISE05]
- gold standard trees derived manually
- less complex, avg. number of internal nodes in gold standard 1.57

Tel8

- originally 487 interfaces of eight domains, about 50% no more accessible
- collected for extraction technique [BEP04]
- provides gold standard in flat representation, no trees

Evaluation (II): Results & Measures



Field

- how well are the labels for leaves extracted
- accuracy given by ratio of number of correctly labelled fields by number of fields

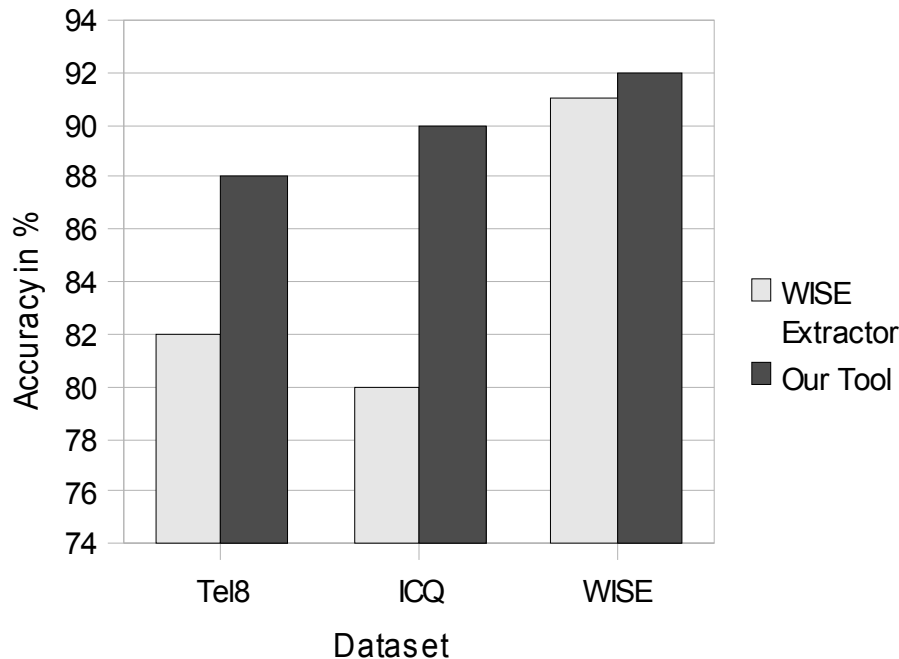
Structure

- how well the groups of fields are identified
- how good the order of fields is obtained
- uses structural tree edit distance
- ignores labels

Overall

- how well the trees and their labels are extracted
- utilizes tree edit distance

Evaluation (III) : Comparison to WISE Extractor



Node Accuracy Measure

- differs from metrics before due to comparability
- ratio of number of fields and internal nodes correctly labelled by the number of nodes
- averaged over all interfaces of dataset

Outline

1. Motivation
2. Concepts
3. Observations on Query Interfaces
4. Extraction Algorithm
5. Evaluation
- 6. Conclusion and Outlook**

Conclusion and Outlook

Conclusion

- query interface extraction technique for Deep Web integration
- visual approach, independent of HTML implementation issues
- represents query interface schemas hierarchically
- parallel to document headings design
- utilizes integration techniques to construct schema tree

Outlook

- handling of images
- integration of semantic tools such as WordNet

References

[WISE05]

Hai He, Weiyi Meng, Clement T. Yu, and Zonghuan Wu. Constructing interface schemas for search interfaces of Web databases. In WISE, 2005.

[BEP04]

Zhen Zhang, Bin He, and Kevin Chen-Chuan Chang. Understanding web query interfaces: best-effort parsing with hidden syntax. In SIGMOD, 2004.

[CLUST04]

Wensheng Wu, Clement Yu, AnHai Doan, and Weiyi Meng. An interactive clustering-based approach to integrating source query interfaces on the deep web. In SIGMOD, 2004.