

## Wrapper Induction: Construct wrappers automatically to extract information from web sources

Hongfei Qu  
Computing Science Department  
Simon Fraser University

CMPT 882 Presentation  
March 28, 2001

## Outline:

- What is wrapper
- Wrapper Induction
- WIEN
- STALKER
- Remaining Questions
- HTML DOM Tree
- Other Related Works
- References

## What is wrapper

- Wrapper is a procedure to extract all kinds of data from a specific web source
- First find a vector of strings to delimit the extracted text
- `<HTML><TITLE>Country Codes</TITLE>  
<BODY><B>Congo</B> <I>242</I><BR>  
<B>Spain</B> <I>34</I><BR>  
<HR><B>END</B></BODY></HTML>`
- To extract pair (country, codes), we find a vector of strings (`<B>`, `</B>`, `<I>`, `</I>`) to distinguish left & right of extracted text.

## What is wrapper

- `execLR(wrapper(<B>, </B>, <I>, </I>), page P):`  
 $m = 0$   
while there are more occurrences in  $P$  of `<B>`  
 $m = m + 1$   
for each  $(lk, rk)$  in  $\{(\text{<B>, </B>}, (\text{<I>, </I>}))\}$   
scan in  $P$  to the next occurrence of  $lk$  in  $P$ ;  
save position as  $bm,k$   
scan in  $P$  to the next occurrence of  $rk$  in  $P$ ;  
save position as  $em,k$   
Return label{...( $bm,1, em,1$ ), ( $bm,2, em,2$ )...}

## Wrapper Induction

- Motivations: hand-coded wrapper is tedious and error-prone. How about web pages get changed?
- Wrapper induction -- automatically generate wrapper --- is a typical machine learning technology.
- Input: a set  $E$  of example pages  $P_n$  and the corresponding label pages  $L_n$
- Output: a wrapper  $w$  such that  $w(P_n) = L_n$

## Wrapper Induction

- Actually we are trying to learn a vector of delimiters, which is used to instantiate some wrapper classes (templates), which describe the document structure
- Free text & Web pages
- A good wrapper induction system should be:
  - Expressiveness: concern how the wrapper handles a particular web site
  - Efficiency: how many samples are needed? How much computational is required?

## WIEN

- First wrapper induction system implemented by U. Washington. Works for both Web page and free text.
- WIEN defines 6 wrapper classes (templates) to express the structures of web sites.
- The simplest and powerful one is LR (left-right) wrapper class. It uses left- and right-hand delimiter to extract the relevant information
- To extract tuples with  $K$  attributes from a set of examples  $E$ , the learning algorithm is:

## WIEN

- Procedure  $learnLR(examples E)$   
for each  $1 \leq k \leq K$   
for each  $u$  in  $Cand_l(k, E)$ : if  $u$  is valid for the  $k$ th attribute in  $E$ , then  $l_k = u$  and terminate the loop  
for each  $1 \leq k \leq K$   
for each  $u$  in  $Cand_r(k, E)$ : if  $u$  is valid for the  $k$ th attribute in  $E$ , then  $r_k = u$  and terminate the loop  
return LR wrapper( $l_1, r_1, \dots, l_k, r_k$ )
- Procedure  $Cand_l(k, E)$  returns candidates for  $l_k$  by enumerating the suffixes of the shortest string occurring to the left of each attribute  $k$  instances

## WIEN

- Procedure  $Cand_r(k, E)$  returns candidates for  $r_k$  by enumerating the prefixes of the shortest string occurring to the right of each attribute  $k$  instances;
- Each wrapper class has a set of validating constraints
- Other wrapper classes:
  - HLRT: add head delimiter  $h$  & tail delimiter  $t$
  - OCLR: using open and close delimiters to indicate the beginning and end of each tuple
  - HOCLRT: combination of HLRT and OCLR
  - N-LR and N-HLRT: handle nested structure
- Combination of 6 classes can handle 70% web sites

## WIEN

- Which wrapper class do we choose for a web site?
- How many examples are required? PAC model  
 $N$ : number of examples;  
 $\epsilon$ : accuracy parameter.  $0 < \epsilon < 1$   
 $\alpha$ : confidence parameter.  $0 < \alpha < 1$   
For a learning wrapper  $W$ , if we want  $error(W) < \epsilon$  with probability at least  $\alpha$ , the PAC model for the LR class is:  
 $N \geq 1/(1-\alpha) * (2K * \ln(R) - \ln(1 - \alpha))$ , where  $R$  is the length of the shortest example.
- A way to terminate the learning procedure
- A loose bound compared with test results

## STALKER

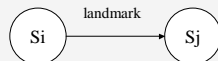
- A wrapper induction project by U. Southern California. Only works for Web page.
- More expressive and efficient than WIEN.
- Treat a web page as a tree-like structure and handle information extraction hierarchically
- Use disjunctions to deal with the variations. Disjunctive rules are ordered lists of individual disjuncts. The wrapper will successively apply each disjunct in the list until it finds one that matches

## STALKER

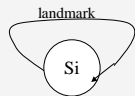
- Landmarks: a sequence of tokens, argument of some functions.  
 $SkipTo(<b>)$ : start from beginning, skip everything until find  $<b>$  landmarks  
 $SkipTo(<b>SkipTo(<I>)$
- These functions represent the rules to extract the information
- Start rule: identify the beginning of an attribute
- End rule: identify the end of an attribute

## STALKER

- These SkipTo() functions represent a finite state machine model
- Extraction rules: get information



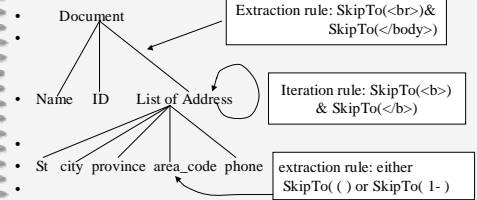
- Iteration rules: handle nested structure



## STALKER

```

<body><p>Name:<b>Hongfei</b></p><p>ID:<b>1111</b></p>
<P>Address:<br><b>4000 Main St, Vancouver, BC, (604)333-3233</b>
</b><br><b>3000 Hastings St, LA, CA, 1-805-486-5675</b></body>
  
```



## STALKER

- Use a sequential covering algorithm
- STALKER(examples)
  - Set **setRule** be empty
  - While there are more examples
    - Get a disjunct **D** by learning examples
    - Remove all examples covered **D**
    - Add **D** into setRule
  - Return **setRule**
- STALKER can handle 90% and more efficient.
- Generate imperfect rules

## Remaining Questions

- Find more expressive model to express document structure
- Select only the informative examples to learn a wrapper.(active learning? Data mining?)
- How to generate label pages automatically instead of hand-markup?

## HTML DOM Tree

- Using a DOM-like tree model on HTML tags



- The navigation methods are similar to XML DOM tree. Only works for web pages.
- Using the tree path to extract information
- Also can follow the document flow like STALKER to extract information
- Get rid of imperfect rules and more efficient

## Other Related Works

- TriAs---html tree
- SOFTMEALY---first use disjunction rule and finite state machine model
- WISK---works for web page and free text, more expressive than WIEN, decision-making is based on limited context. Slower.
- SRV
- CRYSTAL
- RAPIER

## References

- Nicholas Kushmerick, Wrapper Induction: Efficiency and expressiveness, *Artificial Intelligence* 118, 2000
- Ion Muslea, Steven Minton, Craig A. Knoblock, A Hierarchical Approach to Wrapper Induction, *Conference Autonomous Agents*, Seattle, WA, 1999
- S. Soderland, Learning information extraction rules for semi-structured and free text, *Machine Learning* 34, 1999
- C. Hsu, M. Dung, Generating finite-state transducers for semistructured data extraction from the web, *Information Systems* 23, 1998
- M. Bauer, D. Dengler, TriAs—An architecture for trainable information assistants, *Workshop on AI and Information Integration*, Madison, WI, 1998
- D. Freitag, Information extraction from HTML: Application of a general machine learning approach, *AIII-98*, Madison, WI, 1998