DECSAI
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

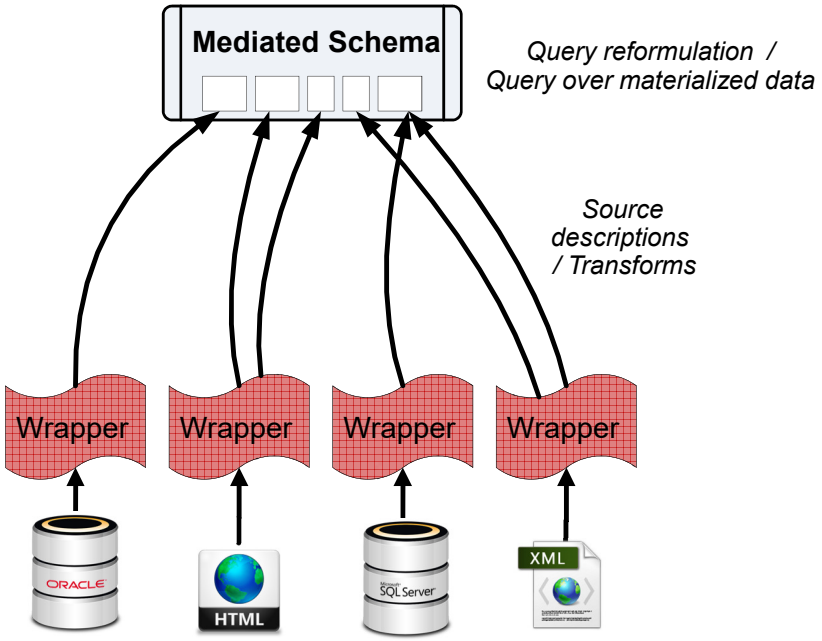
Integración de datos - Wrappers

Fernando Berzal, berzal@acm.org

Integración de datos

- Descripción de fuentes de datos
- Emparejamiento de cadenas [string matching]
- Integración de esquemas
  - Emparejamiento de esquemas [schema matching]
  - Correspondencias entre esquemas [schema mapping]
  - Gestión de modelos
- Emparejamiento de datos [data matching]
- Wrappers
  - Construcción manual
  - Construcción automática
- Apéndice: Procesamiento de consultas

Wrappers



Wrappers

Problema

Tenemos acceso a fuentes de datos
que consisten en conjuntos de páginas web:

Cada fuente de datos S muestra
datos estructurados de acuerdo a un esquema T_S
utilizando un formato F_S en sus páginas web.

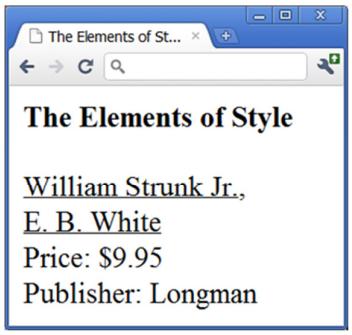
Un wrapper W extrae los datos estructurados
de las páginas web de la fuente de datos S.

Wrappers

EJEMPLOS

(a) *countries.com*

(b) *easycalls.com*



(c) *greatbooks.com*

Muy comunes en la Web:

Un script crea la página HTML que contiene las tuplas devueltas como resultado a una consulta del usuario que se ejecuta sobre una base de datos.

Wrappers

Formalización

Un wrapper W es una tupla (T_W, E_W):

- T_W es el esquema de destino [target schema]
- E_W es el programa de extracción que utiliza el formato F_S para extraer de cada página una instancia de acuerdo al esquema T_W.

Wrappers

EJEMPLO

countries.com

T_{W2} = (country, capital)

T_{W*} = (country, capital, population, continent)

Wrappers

El problema de la construcción de wrappers
a.k.a. aprendizaje de wrappers [wrapper learning]

Construir $W = (T_W, E_W)$ inspeccionando las páginas de S.

Variantes

- Dado el esquema T_W,
  construir el programa de extracción E_W.
- Sin un esquema dado T_W,
  identificar el esquema T_S y construir E_W.

Wrappers

El problema de la construcción de wrappers

a.k.a. aprendizaje de wrappers [wrapper learning]

Problema 1: Identificar el esquema T_S es muy difícil.

- Se interpreta cada página de S como una cadena generada por una gramática G.
- Se aprende G a partir de un conjunto de páginas de S y se utiliza G para inferir T_S.

Wrappers

El problema de la construcción de wrappers

a.k.a. aprendizaje de wrappers [wrapper learning]

Problema 1: Identificar el esquema T_S es muy difícil.

EJEMPLO

Las páginas de countries.com
pueden generarse con la gramática regular:

```
R = <html>.+?<hr><br>(.+?)
    <br>Capital: (.+?)
    <br>Population: (.+?)
    <br>Continent: (.+?)</html>
```

Wrappers

El problema de la construcción de wrappers

a.k.a. aprendizaje de wrappers [wrapper learning]

Problema 1: Identificar el esquema T_S es muy difícil.

Inferir una gramática a partir de ejemplos positivos (páginas de S) es un problema difícil:

- Las gramáticas regulares no pueden identificarse correctamente sólo a partir de ejemplos positivos.
- Aunque dispusiésemos de ejemplos negativos, no existe un algoritmo eficiente que identifique una gramática "razonable" (p.ej. mínima).

Wrappers

El problema de la construcción de wrappers

a.k.a. aprendizaje de wrappers [wrapper learning]

Problema 1: Identificar el esquema T_S es muy difícil.

En la práctica, se consideran sólo gramáticas regulares relativamente simples que codifican esquemas con simples tuplas (tal vez anidadas) para que el problema sea tratable.

Incluso así, hay que recurrir a heurísticas para acotar el espacio de búsqueda de esquemas candidatos (lo que puede conducir a resultados incorrectos).

Wrappers

El problema de la construcción de wrappers

a.k.a. aprendizaje de wrappers [wrapper learning]

Problema 2: Construir el programa de extracción E_W.

- Idealmente, E_W debería ser Turing-completo (máxima capacidad expresiva) pero no es práctico.
- Se escoge E_W utilizando un modelo computacional restringido: sólo se aprende un conjunto limitado de parámetros del modelo.
- Incluso así, el problema sigue siendo muy difícil.

Wrappers

El problema de la construcción de wrappers

a.k.a. aprendizaje de wrappers [wrapper learning]

Problema 3: Excepciones en el formato de los datos.

- No aparentes al inspeccionar un pequeño conjunto de páginas para crear el wrapper.
- Invalidan las suposiciones realizadas sobre el esquema y/o el formato de los datos.
- Nos obligan a revisar continuamente tanto el esquema de la fuente T_S como el programa de extracción E_W.

Wrappers

El problema de la construcción de wrappers
a.k.a. aprendizaje de wrappers [wrapper learning]

Soluciones habituales

- **Construcción manual** (de T_W y E_W).
- **Aprendizaje** (con esquema).
  Se marcan los atributos de T_w en un conjunto de páginas de S y se aplica un algoritmo de aprendizaje para construir E_W.
- **Construcción automática** (de T_W y E_W).
- **Construcción interactiva**

Wrappers

El problema de la construcción de wrappers
a.k.a. aprendizaje de wrappers [wrapper learning]

- Construcción manual
- Aprendizaje (con esquema)
- Construcción automática (sin esquema)
- Construcción interactiva

Construcción manual

Se examina un conjunto de páginas de S
para crear manualmente tanto el esquema T_W
como el programa de extracción E_W.

Formas de interpretar una página:

- Como un texto (cadena de caracteres).
- Como un árbol DOM [Document Object Model].
- Visualmente (bloques).

Construcción manual

EJEMPLO
Programa en Perl para extraer tuplas (país, prefijo)
del texto de una página web

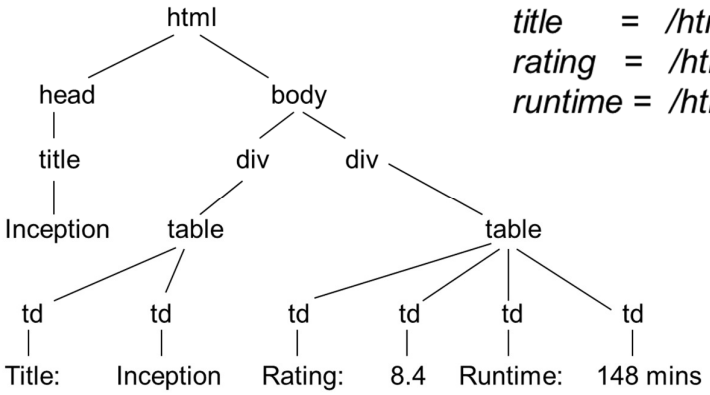

```
#!/usr/bin/perl -w

open(INFILE, $ARGV[0]) or die "can't open file\n";
while ($line = <INFILE>) {
  if ($line =~ m/<B\>(.+?)\<\/B\>\s+?\<I\>(\d+?)\<\/I\>\<BR\>/) {
     print "($1,$2)\n";
  }
}
close(INFILE);
```

Construcción manual

EJEMPLO

Expresiones en XPath para extraer tuplas del árbol DOM asociado a una página web

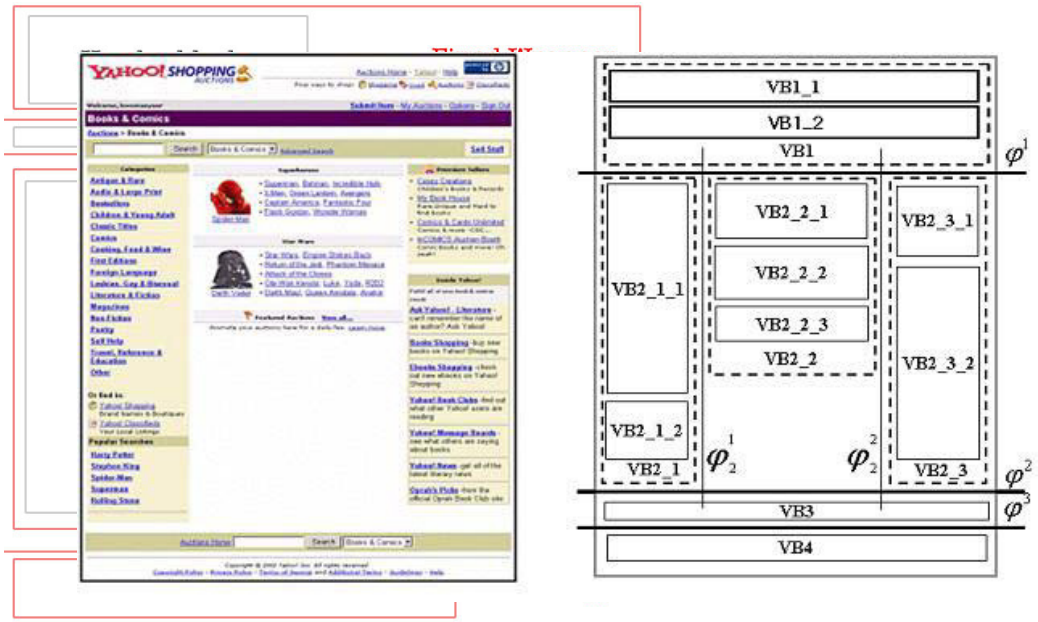


title = /html/body/div[1]/table/td[2]/text()
rating = /html/body/div[2]/table/td[2]/text()
runtime = /html/body/div[2]/table/td[4]/text()

Construcción manual

EJEMPLO

Segmentación visual de los bloques de una página web



Construcción manual

Independientemente del modelo que utilicemos para analizar la página (texto, DOM, bloques), construir manualmente el programa de extracción:

- Puede requerir mucho esfuerzo.
- No es robusto (p.ej. cambios en la fuente de datos).
- Implica un coste elevado de mantenimiento.

Wrappers

El problema de la construcción de wrappers
a.k.a. aprendizaje de wrappers [wrapper learning]

- Construcción manual
- Aprendizaje (con esquema)
- Construcción automática (sin esquema)
- Construcción interactiva

Aprendizaje (con esquema)

- Se considera un conjunto limitado de tipos de wrappers (en comparación con la construcción manual de éstos).

  p.ej. HLRT, Stalker

- Se automatiza la construcción del wrapper utilizando técnicas de aprendizaje automático supervisado:

  Se marcan los atributos de T_w en un conjunto de páginas de la fuente de datos S y se aplica un algoritmo de aprendizaje automático para construir E_W.

Aprendizaje (con esquema)

Construcción del conjunto de entrenamiento

Usualmente, mediante el marcado manual de páginas.

- No requiere conocimientos técnicos específicos.

  p.ej. Crowdsourcing utilizando plataformas como Amazon Mechanical Turk.

- Suele requerir menos trabajo que la creación manual de wrappers.

Aprendizaje (con esquema)

HLRT [Head-Left-Right-Tail]

Lenguaje de alto nivel para la especificación de wrappers:

- Más cómodo que un lenguaje imperativo (menores costes de depuración y mantenimiento).
- Menos expresivo que un lenguaje de programación de propósito general.

IDEA BASE:
Utilizar delimitadores para la extracción de tuplas.

Aprendizaje (con esquema)

HLRT [Head-Left-Right-Tail]


```
<HTML>
<TITLE>Countries in Australia (Continent)</TITLE>
<BODY>
<B>Countries in Australia (Continent)</B><P>
<B>Australia</B>  <I>61</I><BR>
<B>East Timor</B>  <I>670</I><BR>
<B>Papua New Guinea</B>  <I>675</I><BR>
<HR>
<B>Copyright easycalls.com</B>
</BODY>
</HTML>
```

head

data region

tail

Aprendizaje (con esquema)

HLRT [Head-Left-Right-Tail]

Para extraer la tupla (país, prefijo), un wrapper HLRT:

- Elimina el encabezamiento usando <P>.
- Elimina el pie utilizando <HR>.
- Extrae el nombre del país como la cadena en la región de datos que se encuentra entre <B> y </B>.
- Extrae el prefijo telefónico asociado al país como la subcadena de la región de datos entre <I> e </I>.

HLRT (<P>, <HR>, <B>, </B>, <I>, </I>)

Aprendizaje (con esquema)

HLRT [Head-Left-Right-Tail]

Formalmente, un wrapper HLRT que extrae n atributos es una tupla de (2n+2) delimitadores:

$$(h, t, l_1, r_1, \ldots, l_n, r_n)$$

Los wrappers HLRT pueden aprenderse a partir de un conjunto de entrenamiento realizando una búsqueda sobre el espacio de todos los wrappers HLRT posibles.

Aprendizaje (con esquema)

HLRT [Head-Left-Right-Tail]

ALGORITMO DE APRENDIZAJE

- Valores posibles para h (encabezamiento):
  - x_i cadena al comienzo de la página p_i hasta la aparición del primer atributo a_1.
  - $\{x_1..x_m\}$ contiene el h correcto.
- Valores posibles para t (pie):
  - x_i cadena al final de la página p_i después de la aparición del último atributo a_n.
  - $\{x_1..x_m\}$ contiene el t correcto.

Aprendizaje (con esquema)

HLRT [Head-Left-Right-Tail]

ALGORITMO DE APRENDIZAJE

- Valores posibles para cada par (l_i, r_i) :
  - l_i debe ser el sufijo común de todas las cadenas (en las páginas etiquetadas) que terminan justo antes del marcador de a_i.
  - r_i debe ser el prefijo común de todas las cadenas (en las páginas etiquetadas) que empiezan justo después del marcador de a_i.

Aprendizaje (con esquema)

HLRT [Head-Left-Right-Tail]

ALGORITMO DE APRENDIZAJE

Una vez conocidos los posibles valores para los elementos de la tupla **$(h, t, l_1, r_1, \ldots, l_n, r_n)$**, se realiza una búsqueda en el espacio combinado de dichos valores:

- Cada combinación de posibles valores da lugar a un wrapper HLRT candidato.
- En cuanto encontremos un wrapper que extraiga correctamente todos los valores de las páginas etiquetadas, hemos terminado.

Aprendizaje (con esquema)

HLRT [Head-Left-Right-Tail]

Limitaciones prácticas

- Estructura de datos simple (tuplas "planas").

  p.ej. Libro como (título, autores, precio) donde autores es una lista de tuplas (nombre, apellidos)

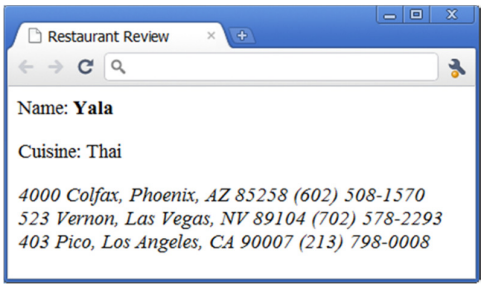
- Extracción utilizando sólo delimitadores.

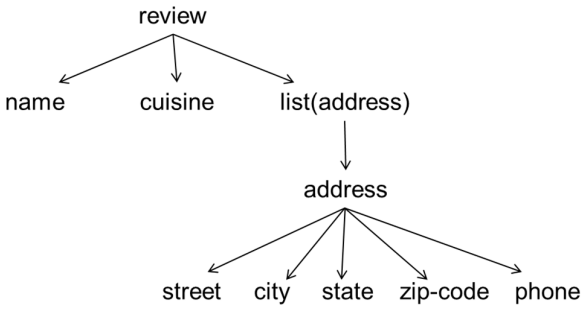
  p.ej. ¿Código postal de "UGR, Granada 18071"?

Aprendizaje (con esquema)

Stalker

TUPLAS ANIDADAS





Situación muy común en la web…

Aprendizaje (con esquema)

Stalker

TUPLAS ANIDADAS

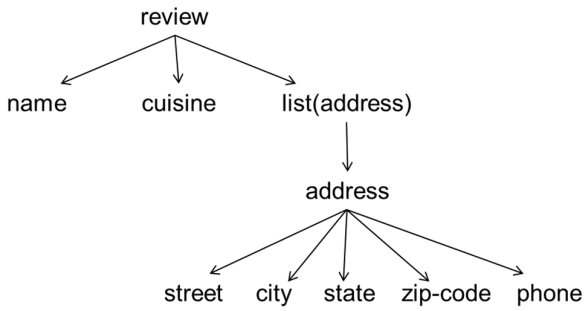
N: Conjunto de todos los esquemas de tuplas anidadas

- El esquema correspondiente a una cadena que representa a un dato pertenece a N.
- Si $T_1,..,T_n$ pertenecen a N, el esquema de la tupla $(T_1,..,T_n)$ también pertenece a N.
- Si T pertenece a N, el esquema de la lista <T> también pertenece a N.

Aprendizaje (con esquema)

Stalker

TUPLAS ANIDADAS



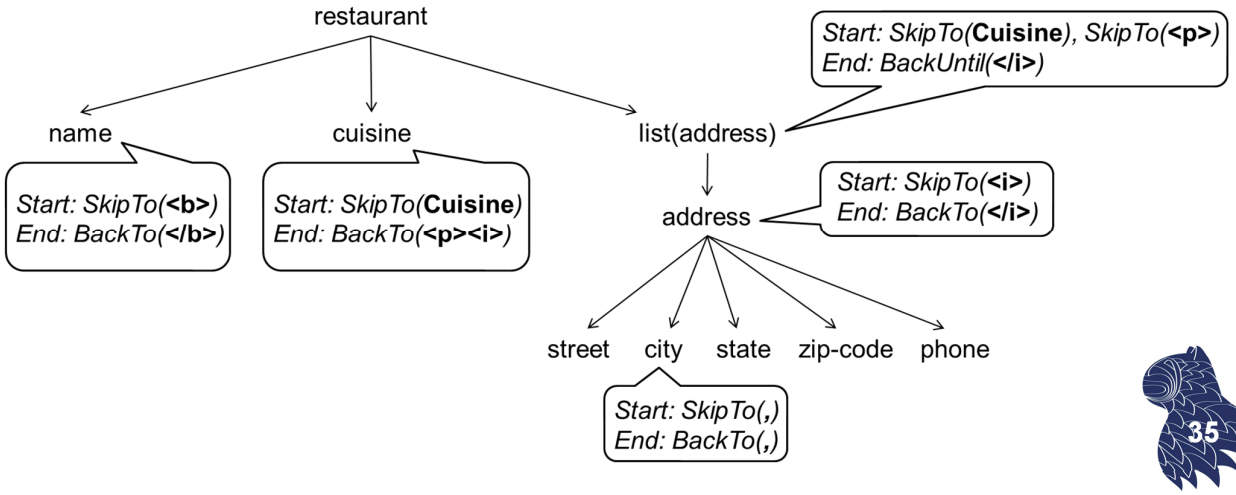
Un esquema de tuplas anidadas puede representarse como un árbol:

- Las hojas son las cadenas.
- Los nodos internos son tuplas o listas.

Aprendizaje (con esquema)

Stalker

MODELO DE WRAPPER: Un wrapper Stalker es un esquema de tuplas anidadas en forma de árbol con reglas en cada nodo para indicar cómo extraer los valores de ese nodo.

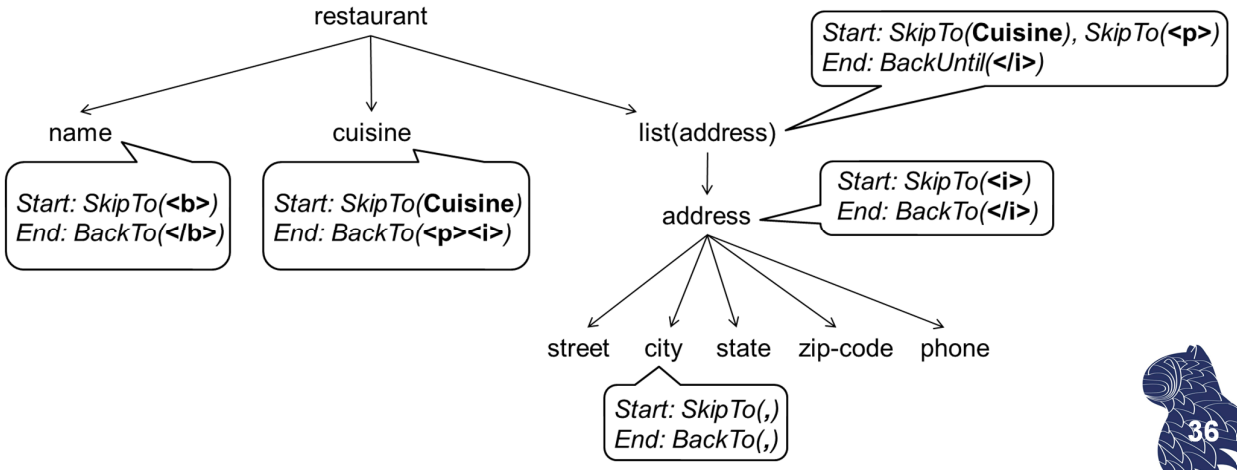


Aprendizaje (con esquema)

Stalker

EJEMPLO

```
<p> Name: <b>Yala</b><p>Cuisine: Thai<p>
<i>4000 Colfax, Phoenix, AZ 85258 (602) 508-1570</i><br>
<i>523 Vernon, Las Vegas, NV 89104 (702) 578-2293</i><br>
<i>403 Pico, LA, CA 90007 (213) 798-0008</i>
```



Aprendizaje (con esquema)

Stalker

REGLAS DE EXTRACCIÓN

- Reglas (contexto: secuencia de comandos)
- Contexto: `Start`, `End`…
- Comandos `SkipTo(marca)`, donde `marca` es una secuencia de tokens y comodines (p.ej. `Punctuation`, `HTMLTag`…), i.e. una expresión regular restringida.
- Los comandos de las reglas se ejecutan secuencialmente (se consume texto hasta encontrar una cadena de entrada que coincida con la marca).

Aprendizaje (con esquema)

Stalker

ALGORITMO DE APRENDIZAJE

Entrada

- Esquema de tuplas anidadas en forma de árbol.
- Conjunto de páginas previamente etiquetadas.

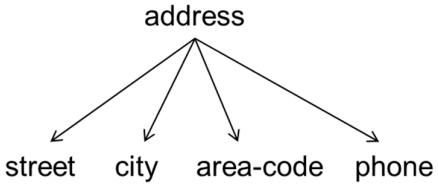
Salida:

- Reglas para los nodos del árbol (start & end rules).

Aprendizaje (con esquema)

Stalker

ALGORITMO DE APRENDIZAJE



E_1: 513 Pico, Venice, Phone: 1-800-555-1515

E_2: 90 Colfax, Palms, Phone: (818) 508-1570

E_3: 523 First St., LA, Phone: 1-888-578-2293

E_4: 403 La Tijera, Watts, Phone: (310) 798-0008

Algoritmo de recubrimiento secuencial (iterativamente, se descubren reglas que cubran subconjuntos de ejemplos del conjunto de entrenamiento):

- 1ª iteración: R_1 = SkipTo(( ) cubre E_2 y E_4.
- 2ª iteración: R_7 = SkipTo(-) cubre E_2 y E_4.

Resultado final: Disyunción de las reglas encontradas.

Aprendizaje (con esquema)

Stalker

ALGORITMO DE APRENDIZAJE

El algoritmo de recubrimiento secuencial puede considerar un conjunto enorme de posibles reglas, p.ej. para descubrir R_7 = SkipTo(-).

Wildcards: *Anything, Numeric, AlphaNumeric, Alphabetic, Capitalized, AllCaps, HTMLTag, nonHTML, Puctuation*

R_7: *SkipTo(-* *****)*
R_8: *SkipTo(Punctuation* *****)*
R_9: *SkipTo(Anything* *****)*
R_{10}: *SkipTo*(**Venice**) *SkipTo*(****)
R_{11}: *SkipTo*(****) *SkipTo*(****)
R_{12}: *SkipTo*(**:**) *SkipTo*(****)
R_{13}: *SkipTo*(**-**) *SkipTo*(****)
R_{14}: *SkipTo*(**,**) *SkipTo*(****)
R_{15}: *SkipTo*(**Phone**) *SkipTo*(****)
R_{16}: *SkipTo*(**1**) *SkipTo*(****)
R_{17}: *SkipTo*(*Numeric*) *SkipTo*(****)
R_{18}: *SkipTo*(*Punctuation*) *SkipTo*(****)
R_{19}: *SkipTo*(*HTMLTag*) *SkipTo*(****)
R_{20}: *SkipTo*(*AlphaNum*) *SkipTo*(****)
R_{21}: *SkipTo*(*Alphabetic*) *SkipTo*(****)
R_{22}: *SkipTo*(*Capitalized*) *SkipTo*(****)
R_{23}: *SkipTo*(*NonHTML*) *SkipTo*(****)
R_{24}: *SkipTo*(*Anything*) *SkipTo*(****)

Aprendizaje (con esquema)

- HLRT y Stalker pueden verse como modelos de autómatas finitos.
- Stalker proporciona un modelo más general que HLRT (HLRT = tuplas anidadas, pero sin anidamiento).
- Al imponer una estructura rígida, se hace tratable el problema del aprendizaje: se transforma un problema general en uno mucho más sencillo con un conjunto relativamente pequeño de parámetros (delimitadores o reglas de extracción).
- Aún así, el aprendizaje es muy difícil: espacio de búsqueda enorme que requiere el uso de heurísticas.

Wrappers

El problema de la construcción de wrappers
a.k.a. aprendizaje de wrappers [wrapper learning]

- Construcción manual
- Aprendizaje (con esquema)
- Construcción automática (sin esquema)
- Construcción interactiva

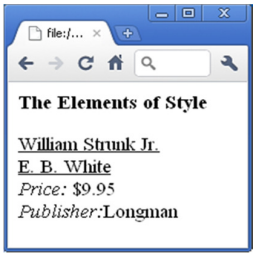
Construcción automática

Aprendizaje de wrappers sin esquema

Dado un conjunto de páginas de una fuente S,
se examinan similitudes y diferencias entre las páginas
para inferir automáticamente su esquema T_S
y construir el programa de extracción E_W
que extrae datos de acuerdo a T_S.

Construcción automática

Aprendizaje de wrappers sin esquema



AB+C?D

<HTML><B>#PCDATA</B><P>(<U>#PCDATA</U>
)+
(<I>Price:</I>#PCDATA
)?<I>Publisher:</I>#PCDATA
</HTML>

The Elements of Style	William Strunk Jr.	$9.95	Longman
	E. B. White		
The Snow of Kilimanjaro	Ernest Hemingway		Scribner

Construcción automática

RoadRunner

RoadRunner modela el esquema T_S de la fuente S como un esquema de tuplas anidadas:

- Se admiten elementos opcionales, p.ej. C en AB+C?D.
- No se admiten disyunciones (explosión combinatoria).

RoadRunner modela el programa de extracción E_W como una expresión regular que extrae los atributos de T_S.

<HTML><B>#PCDATA</B><P>(<U>#PCDATA</U>
)+
(<I>Price:</I>#PCDATA
)?<I>Publisher:</I>#PCDATA
</HTML>

Construcción automática

RoadRunner

ALGORITMO DE APRENDIZAJE

Dado un conjunto de páginas P = {$\mathbf{p}_1$, …, $\mathbf{p}_n$},
examinar P para inferir $\mathbf{E}_W$ y,
a continuación, inferir $\mathbf{T}_S$ de $\mathbf{E}_W$

Para inferir $\mathbf{E}_W$, se emplea un algoritmo iterativo:

- Se inicializa $\mathbf{E}_W$ con la página $\mathbf{p}_1$
  (puede verse como una expresión regular)
- Se generaliza $\mathbf{E}_W$ para que también acepte $\mathbf{p}_2$…
- El $\mathbf{E}_W$ final acepta todas las páginas de P.

Construcción automática

RoadRunner

ALGORITMO DE APRENDIZAJE

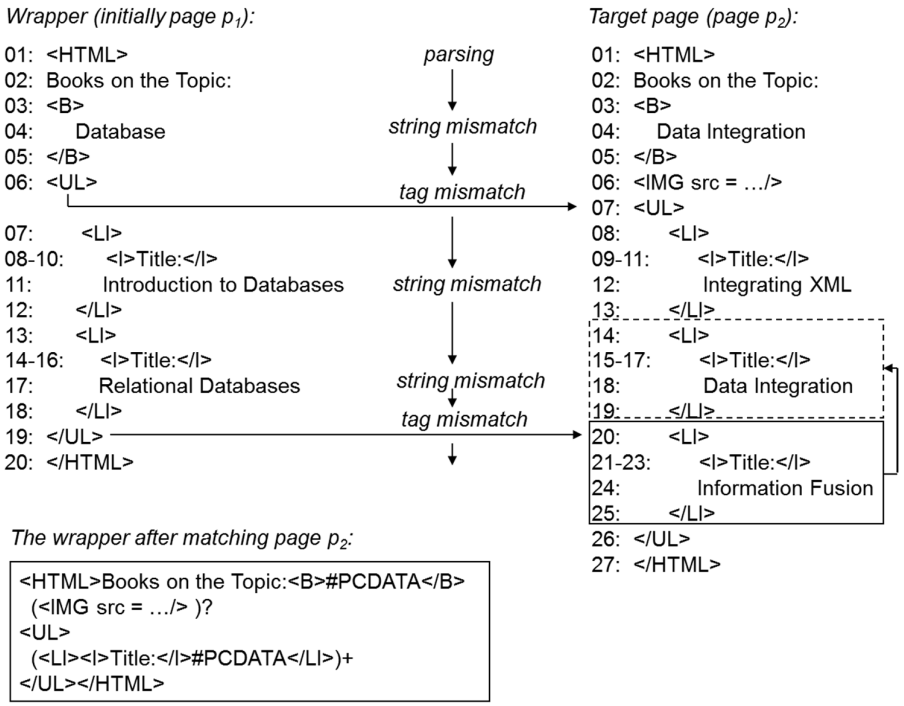
Generalización

Inicializado $\mathbf{E}_W$ para la página $\mathbf{p}_1$
se generaliza $\mathbf{E}_W$ para que también acepte $\mathbf{p}_2$

- Tokenización
  (texto convertido en una secuencia de tokens).
- Comparación de las páginas, desde el primer token.
- Eventualmente, los tokens no coincidirán:
  - Cadenas diferentes (p.ej. "DDSI" vs. "ISI").
  - Tags HTML diferentes (p.ej. <UL> vs. <IMG…>)

Construcción automática

RoadRunner



Construcción automática

RoadRunner

ALGORITMO DE APRENDIZAJE

Generalización

Si las etiquetas no coinciden

- Puede tratarse de un iterador (</UL> vs. <LI>) y, en ese caso, se generaliza para incorporar el iterador (+).
- En caso contrario, puede tratase de un elemento opcional (<IMG…>), que usamos para generalizar (?).

Es importante considerar los iteradores antes que los elementos opcionales.

Construcción automática

RoadRunner

ALGORITMO DE APRENDIZAJE

Generalización

¡OJO! La resolución de iteradores involucra recursividad.

```
<LI>
  Information Fusion
  <B>David Smith</B>
  <B>Jane Lee</B>
</LI>
```

```
<LI>
  Data Integration
  <B>James Madison</B>
</LI>
```


Construcción automática

RoadRunner

ALGORITMO DE APRENDIZAJE

El espacio de búsqueda es enorme
(tiempo exponencial en la fase de generalización).

Uso de heurísticas para reducir el tiempo de ejecución:

- Número de opciones en cada decisión (top k).
- Sin backtracking para ciertos tipos de decisiones.
- Se ignoran ciertos patrones de iteradores y elementos opcionales, por considerarlos altamente improbables.

Wrappers

El problema de la construcción de wrappers
a.k.a. aprendizaje de wrappers [wrapper learning]

- Construcción manual
- Aprendizaje (con esquema)
- Construcción automática (sin esquema)
- Construcción interactiva

Construcción interactiva

Las técnicas automáticas
- son muy costosas computacionalmente y
- utilizan heurísticas para guiar el proceso de búsqueda,

pero las heurísticas no son perfectas
y los resultados que se obtienen son frágiles.

Enfoque alternativo
- Se comienza con pocas (o ninguna) entrada, hasta que aparece cierta incertidumbre.
- Se consulta con el usuario la forma de resolver los conflictos antes de proseguir la búsqueda.
- El proceso se repite iterativamente hasta obtener el wrapper deseado.

Construcción interactiva

Etiquetado interactivo con Stalker

APRENDIZAJE ACTIVO: CO-TESTING

Modificación de Stalker para que le pida al usuario etiquetar páginas durante la búsqueda (no antes):

- El usuario etiqueta una (o varias) páginas.
- Stalker crea un wrapper inicial.
- Intercala la búsqueda con la opinión del usuario.

¿Qué pagina se elige para que el usuario la etiquete?

- Se mantienen dos wrappers candidatos.
- Se escogen páginas "problemáticas" en los que los wrappers difieran.

Construcción interactiva

Etiquetado interactivo con Stalker

APRENDIZAJE ACTIVO: CO-TESTING

1. Etiquetado inicial:

   Name:Savory<p>Phone: (608) 263-4567 <p>Fax:(608) 523-4917

2. Aprendizaje de dos wrappers
   - Forward rule $\mathbf{R}_1$: SkipTo(Phone:)
   - Backward rule $\mathbf{R}_2$: BackTo(Fax), BackTo(()
3. Se aplican los wrappers en un conjunto de páginas no etiquetadas y se marcan como problemáticas las páginas en las que la extracción no coincida.
4. El usuario etiqueta una página problemática y se vuelve al paso 2 hasta que no haya más errores.

Construcción interactiva

Poly

Co-testing, como en la versión modificada de Stalker.

- Se mantienen múltiples wrappers (en vez de dos).
- El usuario identifica los resultados correctos.
- Se usa un modelo DOM/visual (en vez de cadenas).

Generación de wrappers

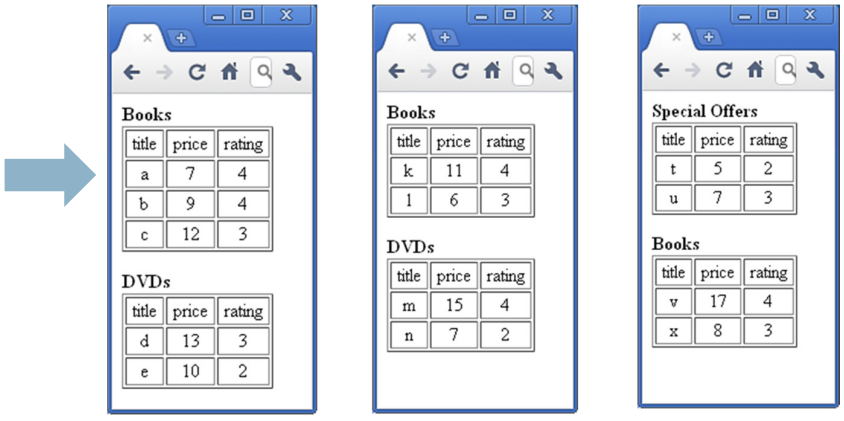
- Se convierte la página en un árbol DOM.
- Se identifican los nodos que corresponden a los atributos seleccionados por el usuario.
- Se crean expresiones XPath.

Construcción interactiva

Poly

FASE 1: INICIALIZACIÓN

Múltiples tuplas por página, de las cuales se quiere extraer un subconjunto. El usuario marca alguna de una de las páginas…



Books

title	price	rating
a	7	4
b	9	4
c	12	3

DVDs

title	price	rating
d	13	3
e	10	2

Books

title	price	rating
k	11	4
l	6	3

DVDs

title	price	rating
m	15	4
n	7	2

Special Offers

title	price	rating
t	5	2
u	7	3

Books

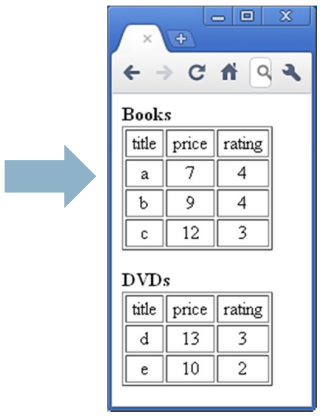
title	price	rating
v	17	4
x	8	3

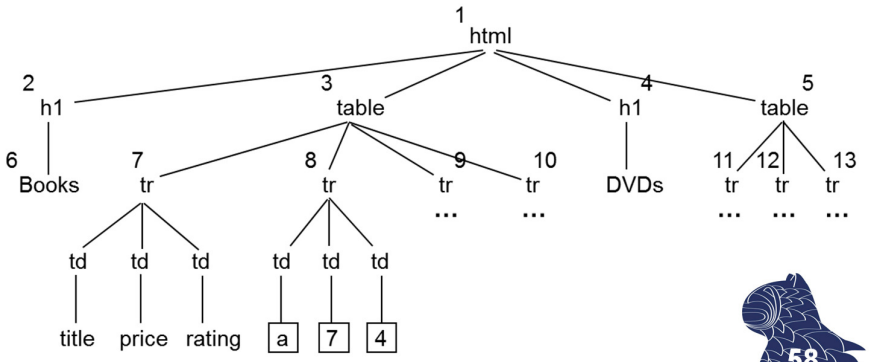
Construcción interactiva

Poly

FASE 2: GENERACIÓN DE MÚLTIPLES WRAPPERS

Se generan múltiples wrappers, cada uno de los cuales extrae tuplas que incluyen la seleccionada por el usuario.





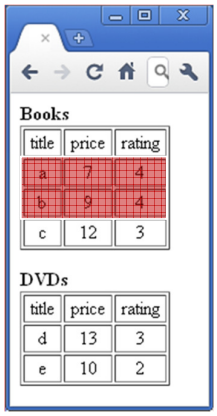
Construcción interactiva

Poly

FASE 3: IDENTIFICACIÓN DEL RESULTADO CORRECTO

Se muestran los resultados proporcionados por los distintos wrappers y el usuario selecciona los que proporcionan el resultado correcto (los demás se descartan).

Los wrappers aún no descartados proporcionan la respuesta correcta para la página actual, que ya no es útil para discriminar…

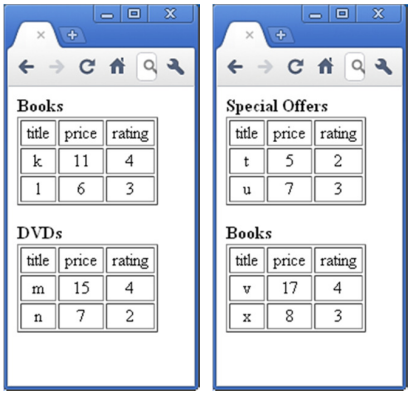


Construcción interactiva

Poly

FASE 4: EVALUACIÓN EN OTRAS PÁGINAS

Se aplican los distintos wrappers sobre las páginas restantes (no etiquetadas) y se comprueba en cuáles proporcionan resultados diferentes.



Si se encuentra alguna diferencia, se vuelve a la fase 3 usando esa página.

Si no hay diferencias en las páginas no etiquetadas, cualquiera de los wrappers candidatos nos vale.

Construcción interactiva

Lixto

- Se crean reglas de extracción visualmente, marcando páginas con ayuda de cuadros de diálogo.
- Se codifican las reglas de extracción, definidas sobre modelos DOM de las páginas, utilizando un lenguaje similar a Datalog.

¿Cómo se crean las reglas?

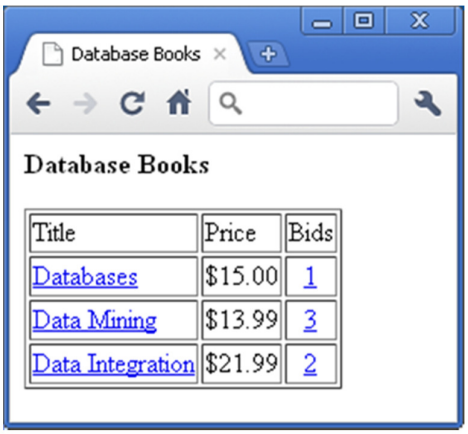
El usuario…

- marca una tupla o valor.
- utiliza cuadros de diálogo para relajar/restringir reglas.
- especifica expresiones regulares.
- hace referencia a conceptos ya definidos en Lixto.

Construcción interactiva

Lixto

REPRESENTACIÓN INTERNA DE LAS REGLAS



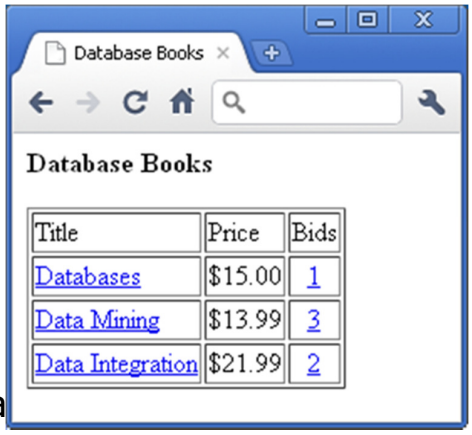
- Regla 1: Libros.
- Regla 2: Título del libro.
- Regla 3: Precio del libro.
- Regla 4: # pujas [bids].

R_1: *book(S,X)* ← *page(S), subelem(S,.table,X)*
R_2: *title(S,X)* ← *book(-,S), subelem(S,(*.td.*.content,[(href,,substr)]),X), notbefore(S,X,.td,100)*
R_3: *price(S,X)* ← *book(-,S), subelem(S,(*.td,[(elementtext,\var[Y].*,regvar)]),X), isCurrency(Y)*
R_4: *bids(S,X)* ← *book(-,S), subelem(S,*.td,X), before(S,X,.td,0,30,Y,-), price(-,Y)*

Construcción interactiva

Lixto

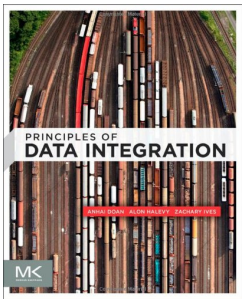
INTERFAZ DE USUARIO



- Regla 1: Libros.
  - El usuario selecciona una tupla.
  - Lixto determina el subárbol DOM correspondiente y crea la regla.
  - Lixto muestra el resultado sobre la propia página y el usuario lo acepta
- Reglas 2-4: Título/precio/#pujas de cada libro.
  - El usuario indica que estas reglas se aplican a las instancias de libros identificadas por la regla 1.
  - El usuario selecciona el atributo particular.
  - Lixto crea una regla y muestra los resultados al usuario.
  - Si la regla es demasiado general, el usuario la refina.

Bibliografía recomendada

- Hai Doan, Alon Halevy & Zachary Ives:
  Principles of Data Integration
  Morgan Kaufmann, 1st edition, 2012.
  ISBN 0124160441
  http://research.cs.wisc.edu/dibook/



Chapter 9: Wrappers