ISSN 0103-9741

Monografias em Ciência da Computação
n° 11/10

First version of a Prototype for Publishing Deep Web Data

Antonio L. Furtado   Simone D. J. Barbosa
Marco A. Casanova   Helena Piccinini

Departamento de Informática

Monografias em Ciência da Computação, No. 11/10 ISSN 0103-9741
Editor: Prof. Carlos José Pereira de Lucena July 2010

First version of a Prototype for Publishing Deep Web Data

Antonio L. Furtado, Simone D. J. Barbosa, Marco A. Casanova, Helena Piccinini

{furtado,simone,casanova}@inf.puc-rio.br
helena.piccinini@gmail.com

Abstract: To make data stored in relational databases accessible to search engines by applying the *W-Ray* method, introduced in a previous paper, a prototype called **WRay1** was developed. Working on views designed according to the ER (Entity-Relationship) modelling principles, **WRay1** endeavours to express the data items of more general interest under the form of template-driven natural language sentences. The sentences are registered on html pages, which are then posted on the Web.

Keywords: deep web, relational databases, views, ER model, SWI-Prolog, ODBC, Oracle, html pages, Google.

Resumo: Para tornar dados armazenados em bancos de dados relacionais acessíveis a mecanismos de busca aplicando o método *W-Ray*, introduzido em um artigo anterior, foi desenvolvido um protótipo, denominado **WRay1**. Operando sobre visões projetadas de acordo com os princípios do modelo ER (Entidades-Relacionamentos), **WRay1** se propõe a expressar os itens de dados de maior interesse sob a forma de frases em linguagem natural, geradas com o auxílio de moldes. As frases são registradas em páginas html, que são em seguida expostas na Rede.

Palavras-chave: web profunda, bancos de dados relacionais, visões, modelo ER, SWI-Prolog, ODBC, Oracle, páginas html, Google.

1. Introduction

Unlike the *Surface Web* of static pages, the *Deep Web* [1] comprises data stored in databases, dynamic pages, scripted pages and multimedia data, among other types of objects. Estimates suggest that the size of the Deep Web greatly exceeds that of the Surface Web – with nearly 92,000 terabytes of data on the Deep Web versus only 167 terabytes on the Surface Web, as of 2003. In particular, Deep Web databases are typically under-represented in search engines due to the technical challenges of locating, accessing, and indexing the databases. Indeed, since Deep Web data is not available as static Web pages, traditional search engines cannot discover data stored in the databases through the traversal of hyperlinks, but rather they have to interact with (potentially) complex query interfaces.

Two basic approaches to access Deep Web data have been proposed. The first approach, called *surfacing*, or *Deep Web Crawl* [16], tries to automatically fill html forms to query the databases. Queries are executed offline and the results are translated to static Web pages, which are then indexed [15]. The second approach, called *federated search*, or *virtual integration* [4, 18], suggests using domain-specific mediators to facilitate access to the databases. Hybrid strategies, extending the previous ones, have also been proposed [21].

Despite recent progress, accessing Deep Web data is still a challenge, for two basic reasons [20]. First, there is the question of scalability. Since the Deep Web is orders of magnitude larger than the Surface Web [1], it may not be feasible to completely index the Deep Web. Second, databases typically are forced to rely on interfaces designed for human users, which complicates the development of software agents to interact with them.

The paper **"W-Ray: A Strategy to Publish Deep Web Geographic Data"**, which is the main reference [0] on which the present report is based, proposes a different approach, called *W-Ray* by analogy with medical X-Ray technology, to publish conventional (as well as geographic data, in vector or raster format – but note that these are not treated in the present report) stored in the Deep Web. The basic idea consists of creating a set of natural language sentences, with a simple structure, to describe Deep Web data, and publishing the sentences in static Web pages, which are then indexed as usual. Employing natural language sentences is convenient for three reasons. First, they lead to Web pages that are acceptable to Web crawlers that consider words randomly distributed in a page as an attempt to manipulate page rank. Second, they facilitate the task of more sophisticated engines that support semantic search based on natural language features [5, 23]. Lastly, the descriptions thus generated are minimally acceptable to human users. The Web pages should be generated following the W3C guidelines [3] and the recommendations published by Google to optimize Web site indexing [9]. The main reference paper [0] illustrates the use of templates over a relational view of the SIDRA database, which the Brazilian Institute of Geography and Statistics (IBGE) publishes on the Web with the help of html forms. It should be observed that SIDRA is not indexed by any conventional search engine.

As an complement to the main reference paper, the present report provides an overview of the first version of a prototype, called **WRay1**, which purports to partially implement the *W-Ray* method. Similarly using a minimal "Geography of Northern Region" example, the prototype's functioning is presented along three stages, covering the selection of the data to be published (section 2), the generation of template-driven natural language sentences (section 3) and the preparation of html pages to be posted on the Web (section 4). Section 5 contains conclusions and topics needing further work. The additional references, included here for convenience, were copied from the main reference paper.

2. Selecting the data to be published

Given a relational database, composed of one or more tables, the first step is to choose the items to be published. Apart from the obvious confidentiality restrictions, one must find out what may or may not be of interest to the prospective users.

A sound practice that, unfortunately, is not always adopted when creating a database is to start with a conceptual design of the information system involved. For this purpose, the Entity-Relationship (ER) model offers suitable guidelines, since it allows to specify real-world objects in terms of *meaningful* schemas, which, at the next design stage, can be mapped into relational tables in a straightforward way. For relational databases that have been designed directly, it is possible to derive the missing conceptual ER schema through a *reverse-engineering* process.

Inspecting the conceptual schema, we must decide which *entities* may be of interest and, for such entities, which properties (in **WRay1** limited to *attributes* and *binary relationships*) should be retained. For our example, the chosen subset of the conceptual schema is displayed below, expressed in the following clausal format:

```
entity(<entity class>,<identifying attribute>).
attribute(<entity class>,<attribute>).
relationship(<relationship class>,[<entity class>,<entity class>).
```

conceptual schema:

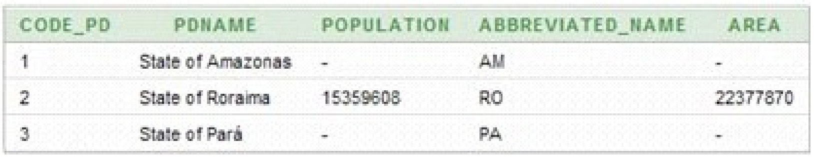
```
entity(political_division,pdname).
entity(populated_place,popname).
entity(waterway,wname).

attribute(political_division,population).
attribute(political_division,abbreviated_name).
attribute(political_division,area).
attribute(populated_place,administration_level).
attribute(populated_place,local_area).
attribute(populated_place,local_population).
attribute(waterway,flow).
attribute(waterway,navigability).

relationship(located_in,[populated_place,political_division]).
relationship(crosses,[waterway,political_division]).
```

From the conceptual schema, one then derives relational tables, which can be regarded as *views* of the original database. To each entity, there corresponds a table, whose columns correspond in turn to the respective attributes. Also, tables are created for the binary relationships, whose columns, called after the participating entity classes, will store the values of their attributes. For the present version we did not take advantage of the reduced representation commonly employed for 1-n relationships. A more important aspect, still currently left aside, is the choice of some standard general or domain-oriented vocabulary, to rename the various items, so as to guide the search engines to a better performance as indicated in the Introduction – we shall refer to that in the Conclusions.

Our example views are shown below, as they were created using the Oracle DBMS and populated with just a few rows, minimally enough to conduct the experiments.

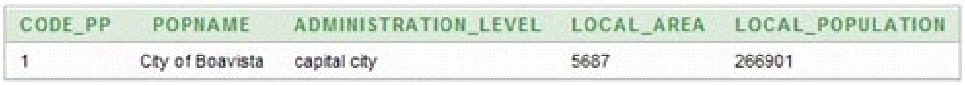
Oracle tables:

POLITICAL_DIVISION



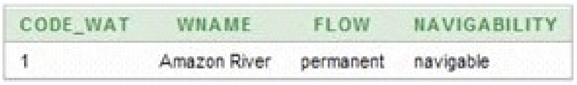
CODE_PD	PDNAME	POPULATION	ABBREVIATED_NAME	AREA
1	State of Amazonas	-	AM	-
2	State of Roraima	15359608	RO	22377870
3	State of Pará	-	PA	-

POPULATED_PLACE



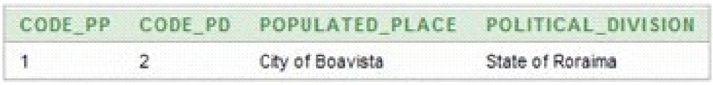
CODE_PP	POPNAME	ADMINISTRATION_LEVEL	LOCAL_AREA	LOCAL_POPULATION
1	City of Boavista	capital city	5687	266901

WATERWAY



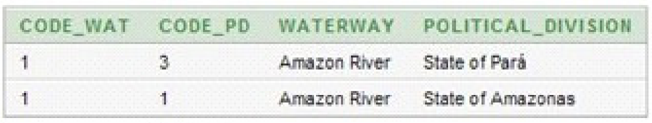
CODE_WAT	WNAME	FLOW	NAVIGABILITY
1	Amazon River	permanent	navigable

LOCATED_IN



CODE_PP	CODE_PD	POPULATED_PLACE	POLITICAL_DIVISION
1	2	City of Boavista	State of Roraima

CROSSES



CODE_WAT	CODE_PD	WATERWAY	POLITICAL_DIVISION
1	3	Amazon River	State of Pará
1	1	Amazon River	State of Amazonas

The SWI-Prolog program empowering the **WRay1** prototype accesses these tables via an **Open Database Connectivity** (**ODBC**) interface. The execution of the command line

```
?- create_queries.
```

compiles a `select` clause for each entity, attribute and relationship present in the conceptual schema, as listed in the sequel:

Compiled selected clauses:

```
flow(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select wname,flow from waterway', C),
      flow(A, B)=..[_|D],
      C=..[row|D].
navigability(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select wname,navigability from waterway', C),
      navigability(A, B)=..[_|D],
      C=..[row|D].
located_in(A, B) :-
```

```
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select populated_place,political_division from located_in', C),
      located_in(A, B)=..[_|D],
      C=..[row|D].
crosses(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select waterway,political_division from crosses', C),
      crosses(A, B)=..[_|D],
      C=..[row|D].
local_area(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select popname,local_area from populated_place', C),
      local_area(A, B)=..[_|D],
      C=..[row|D].
local_population(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select popname,local_population from populated_place', C),
      local_population(A, B)=..[_|D],
      C=..[row|D].
flow(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select wname,flow from waterway', C),
      flow(A, B)=..[_|D],
      C=..[row|D].
navigability(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select wname,navigability from waterway', C),
      navigability(A, B)=..[_|D],
      C=..[row|D].
located_in(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select populated_place,political_division from located_in', C),
      located_in(A, B)=..[_|D],
      C=..[row|D].
crosses(A, B) :-
      odbc_connect('XE', _, [user(prolog), password(prolog), alias(prolog), open(once)]), !,
      odbc_query(prolog, 'select waterway,political_division from crosses', C),
      crosses(A, B)=..[_|D],
      C=..[row|D].
```

By performing selections with these clauses across the ODBC interface, the Prolog program will retrieve all rows in the Oracle tables, when the command line

```
?- forall(fact(F), (F, write(F), nl)).
```

is executed, displaying all the stored data converted into clausal format:

Facts in clause notation:

```
political_division(State of Amazonas)
political_division(State of Roraima)
political_division(State of Pará)
populated_place(City of Boavista)
waterway(Amazon River)
population(State of Amazonas, $null$)
population(State of Roraima, 15359608)
population(State of Pará, $null$)
abbreviated_name(State of Amazonas, AM)
abbreviated_name(State of Roraima, RO)
abbreviated_name(State of Pará, PA)
area(State of Amazonas, $null$)
area(State of Roraima, 22377870)
area(State of Pará, $null$)
```

```
administration_level(City of Boavista, capital city)
local_area(City of Boavista, 5687)
local_population(City of Boavista, 266901)
flow(Amazon River, permanent)
navigability(Amazon River, navigable)
located_in(City of Boavista, State of Roraima)
crosses(Amazon River, State of Pará)
crosses(Amazon River, State of Amazonas)
```

3. Generating template-driven natural language sentences

The same facts above are rendered into template-driven natural language sentences if the command line is rewritten as:

```
?- forall(fact(F), show(F)).
```

in which case the program generates and applies on the fly the appropriate default patterns, all conforming to one of the standard formats below:

There is a(an) <entity class> with <identifying attribute> <attribute value>.
The <attribute> of <identifying attribute value> is <attribute value>.
<identifying attribute value> is related to <identifying attribute value> by <relationship class>.

The generated patterns are internally represented as lists whose items are either constant character strings or variables, to be instantiated with the clause parameters while each clause is matched against the applicable pattern. In our example, the patterns will be:

Patterns in list format:

```
political_division(A)-       ['There is ','a ',political_division,' with ',pdname,' ',A,'.']
populated_place(A)-          ['There is ','a ',populated_place,' with ',popname,' ',A,'.']
waterway(A)-                 ['There is ','a ',waterway,' with ',wname,' ',A,'.']
population(A,B)-             ['The ',population,' of ',A,' is ',B,'.']
abbreviated_name(A,B)-       ['The ',abbreviated_name,' of ',A,' is ',B,'.']
area(A, B)-                  ['The ',area,' of ',A,' is ',B,'.']
administration_level(A,B)-   ['The ',administration_level,' of ',A,' is ',B,'.']
local_area(A, B)-            ['The ',local_area,' of ',A,' is ',B,'.']
local_population(A,B)-       ['The ',local_population,' of ',A,' is ',B,'.']
flow(A, B)-                  ['The ',flow,' of ',A,' is ',B,'.']
navigability(A,B)-           ['The ',navigability,' of ',A,' is ',B,'.']
located_in(A,B)-             [A,' is related to ',B,' by ',located_in,'.']
crosses(A,B)-                [A,' is related to ',B,' by ',crosses,'.']
```

and their matching to the clauses by the `show` predicate, which concatenates the list items after the variables are instantiated so as to compose the sentences, yields the result below:

Facts is template-driven natural language sentences:

```
 There is a political division with pdname State of Amazonas.
 There is a political division with pdname State of Roraima.
 There is a political division with pdname State of Pará.
 There is a populated place with popname City of Boavista.
 There is a waterway with wname Amazon River.
 The population of State of Amazonas is undefined.
 The population of State of Roraima is 15359608.
 The population of State of Pará is undefined.
 The abbreviated name of State of Amazonas is AM.
 The abbreviated name of State of Roraima is RO.
 The abbreviated name of State of Pará is PA.
 The area of State of Amazonas is undefined.
 The area of State of Roraima is 22377870.
 The area of State of Pará is undefined.
 The administration level of City of Boavista is capital city.
 The local area of City of Boavista is 5687.
 The local population of City of Boavista is 266901.
 The flow of Amazon River is permanent.
 The navigability of Amazon River is navigable.
 City of Boavista is related to State of Roraima by located in.
 Amazon River is related to State of Pará by crosses.
 Amazon River is related to State of Amazonas by crosses.
```

If, on the one hand, it is convenient to conform to a uniform standard to ease the job of the search engines, on the other hand a user may find desirable to express certain facts in a more idiomatic and fluent style. To cater to this preference, **WRay1** supplies a dialogue-based feature, wherein the user is shown the default sentence format and asked to indicate the new desired format. The resulting user-templates will be represented as lists, exactly as the default ones, but, once again, users need not bother with internal representation details.

As an example, suppose a user wishes to combine four sorts of facts:

```
political_division(Pd)
crosses(Wa,Pd)
flow(Wa,Fl)
navigability(Wa,Na)
```

The command line

```
?- show((political_division(P), crosses(R, P), flow(R, S),
   navigability(R, V))).
```

would yield two results, each consisting of four sentences:

```
There is a political division with pdname State of Amazonas. Amazon River
is related to State of Amazonas by crosses. The flow of Amazon River is
permanent. The navigability of Amazon River is navigable.
There is a political division with pdname State of Pará. Amazon River is
related to State of Pará by crosses. The flow of Amazon River is
permanent. The navigability of Amazon River is navigable.
```

Now suppose the user finds more palatable to express that information in single sentences, removing the obvious redundancies. A dialogue to achieve this purpose ensues, dealing separately with each of the four facts. The user's replies appear in boldface.

Dialogue to create user-defined template:

```
?- new_template(political_division).
```

```
Default template:
There is a political division with pdname id-political division

Please, type sentence with id-political_division
my choice: The State of id-political_division

?- new_template(crosses).

Default template:
id-waterway is related to id-political division by crosses

Please, type sentence with both id-waterway and id-political division
my choice: is crossed by the id-waterway,

?- new_template(flow).

Default template:
The flow of id-waterway is val-flow

Please, type sentence with both id-waterway and val-flow
my choice: which is val-flow

?- new_template(navigability).

Default template:
The navigability of id-waterway is val-navigability

Please, type sentence with both id-waterway and val-navigability
my choice: and val-navigability.
```

As a consequence, four user patterns are produced, with the internal representation below:

```
utemplate(political_division(A), ['The State of ', A])
utemplate(crosses(A, _), ['is crossed by the ', A, ','])
utemplate(flow(_, A), ['which is ', A])
utemplate(navigability(_, A), ['and ', A, '.'])
```

after which the same command line

```
?- show((political_division(P), crosses(R, P), flow(R, S),
   navigability(R, V))).
```

yields:

```
The State of Amazonas is crossed by the Amazon River, which is permanent
and navigable.
The State of Pará is crossed by the Amazon River, which is permanent and
navigable.
```

Until now we have been assuming screen-output. The Prolog program can instead be directed to produce text output on a txt file. Just as the file is being opened for output, a list of keywords is recorded, enumerating all items is the conceptual schema, optionally preceded by words indicated by the user.

In an attempt to provide an example, we asked for the facts concerning the existence of political divisions, populated places and waterways to be recorded using the default patterns. After this, we caused the creation of the user templates mentioned before (which can be turned off or reactivated, by `remove_uts` and `restore_uts`), and then performed the command line (with `show` replaced by `w_show`, as required to redirect the output). See below the resulting contents of the file.

Contents of the txt file:

```
keywords: Brasil, Brazil, geography, northern region, political division,
populated place, waterway, pdname, popname, wname, population,
abbreviated name, area, administration level, local area, local
population, flow, navigability, located in, crosses.

There is a political division with pdname State of Amazonas.
There is a political division with pdname State of Roraima.
There is a political division with pdname State of Pará.

There is a populated place with popname City of Boavista.

There is a waterway with wname Amazon River.

The State of Amazonas is crossed by the Amazon River, which is permanent
and navigable.
The State of Pará is crossed by the Amazon River, which is permanent and
navigable.
```

4. Posting html pages on the Web

Arguably html is a more adequate file format than txt for publishing on the Web. Accordingly, we proceeded to manually reformat the txt file above to compose what we called the Details page. Besides introducing various html tags, we added, as a heading:

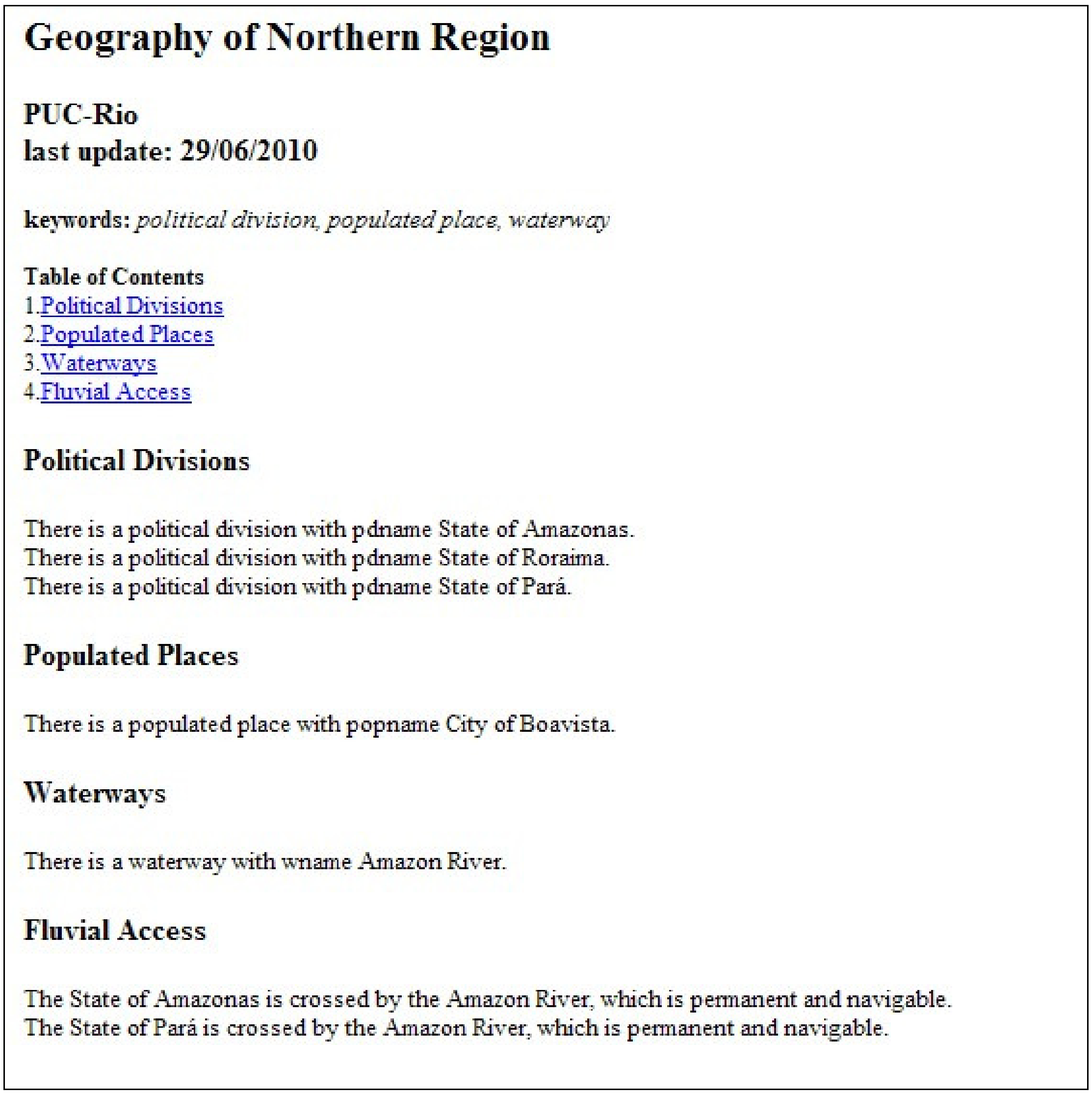
- a title: Geography of Northern Region
- the acronym of our university, to certify the provenance of the document
- the occasion of the last update

and we kept some of the keywords of the txt file. Finally we included a table of contents, with reference tags linking to anchors placed at the beginning of each of the four sets of sentences, thus separating them into distinct *sections*, called:

- Political divisions
- Populated places
- Waterways
- Fluvial access

The Details page is found at: http://www-di.inf.puc-rio.br/~furtado/geo.htm

The html Details page posted on the Web - created from the txt file:



Geography of Northern Region

PUC-Rio
last update: 29/06/2010

keywords: *political division, populated place, waterway*

Table of Contents
1.Political Divisions
2.Populated Places
3.Waterways
4.Fluvial Access

Political Divisions

There is a political division with pdname State of Amazonas.
There is a political division with pdname State of Roraima.
There is a political division with pdname State of Pará.

Populated Places

There is a populated place with popname City of Boavista.

Waterways

There is a waterway with wname Amazon River.

Fluvial Access

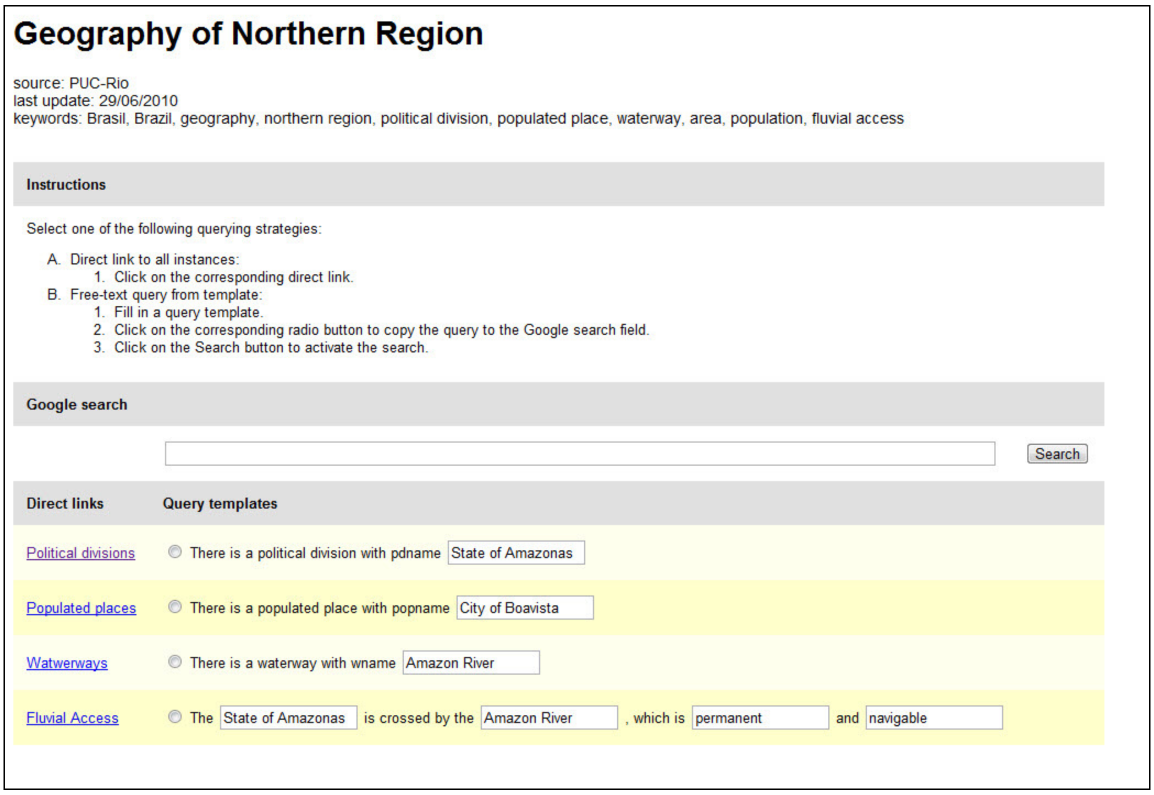
The State of Amazonas is crossed by the Amazon River, which is permanent and navigable.
The State of Pará is crossed by the Amazon River, which is permanent and navigable.

In addition to this Details page, containing all the available detailed information, we provided what we called the Top page, also in html format, with the same heading and keywords. From this page, two kinds of queries can be posed. Both kinds are associated with the section titles, each one exemplified with the first contained sentence:

- direct link: to access the Details page at the beginning of the section, by clicking on the section's title
- free query from template: to submit to Google a query template, possibly after modifying the input boxes in the example sentence, by clicking on the respective radio button and then on the search button placed above

The Top page is found at: http://www-di.inf.puc-rio.br/~furtado/geo_search.htm

The Top html page:


Geography of Northern Region

source: PUC-Rio
last update: 29/06/2010
keywords: Brasil, Brazil, geography, northern region, political division, populated place, waterway, area, population, fluvial access

Instructions

Select one of the following querying strategies:

A. Direct link to all instances:
   1. Click on the corresponding direct link.
B. Free-text query from template:
   1. Fill in a query template.
   2. Click on the corresponding radio button to copy the query to the Google search field.
   3. Click on the Search button to activate the search.

Google search

Direct links	Query templates
Political divisions	There is a political division with pdname [State of Amazonas]
Populated places	There is a populated place with popname [City of Boavista]
Watwerways	There is a waterway with wname [Amazon River]
Fluvial Access	The [State of Amazonas] is crossed by the [Amazon River] , which is [permanent] and [navigable]

The first option is intended, of course, to link directly to the Details page. If one must examine all facts in a section, this is the appropriate choice. The second option, however, may be preferable if one wishes to know about a single specific fact. For instance, suppose the user wants to learn whether 'Roraima' is a political division, and if so, also wants to learn whether it is now called a 'State' (it used to be a 'Territory'). The user would then: replace the contents of the input box of the first query template by '* Roraima', click on the radio button, observe that the Google search box now reads

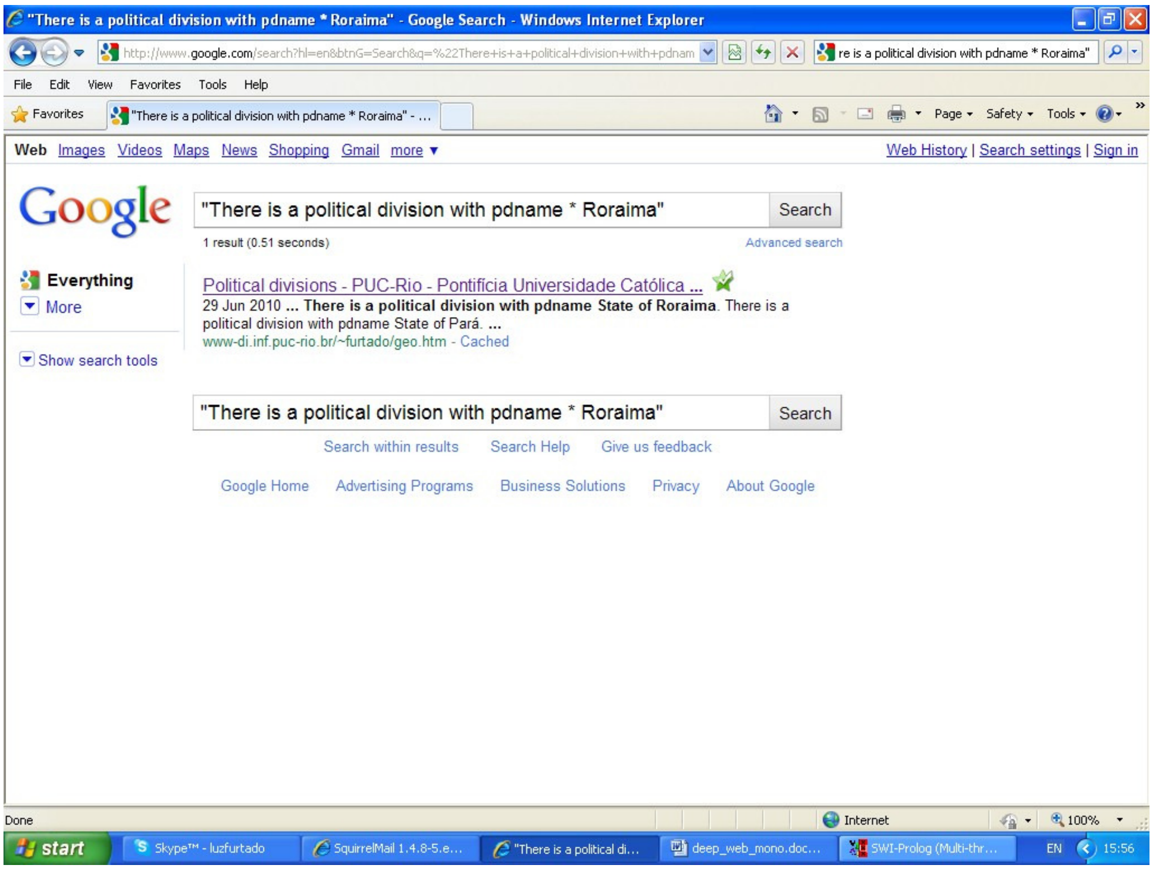
"There is a political division with pdname * Roraima"

and, by clicking on the search button, be led to a Google answer page, where precisely the reply that the user was looking for:

There is a political division with pdname State of Roraima.

appears in a selected *snippet*, conveniently highlighted in boldface as shown in the following printsccreen – with the most desirable consequence that there is no need to access and go through the Details page.

Answer page displayed by Google:


"There is a political division with pdname * Roraima" - Google Search - Windows Internet Explorer
http://www.google.com/search?hl=en&btnG=Search&q=%22There+is+a+political+division+with+pdnam
File Edit View Favorites Tools Help
Web Images Videos Maps News Shopping Gmail more
Web History | Search settings | Sign in
Everything
More
Show search tools
Search
Advanced search
Search within results Search Help Give us feedback
Google Home Advertising Programs Business Solutions Privacy About Google
Done
Internet
100%
start

"There is a political division with pdname * Roraima"

1 result (0.51 seconds)

Political divisions - PUC-Rio - Pontifícia Universidade Católica ...
29 Jun 2010 ... **There is a political division with pdname State of Roraima**. There is a political division with pdname State of Pará. ...
www-di.inf.puc-rio.br/~furtado/geo.htm - Cached

In fairness, however, we must alert the reader that hitting upon a snippet with the desired answer is subject to circumstances related to the Google processes which we still ignore[1]. To give a negative example, the query formulated by altering the fourth template to:

"The State of Pará is crossed by the * River, which is * and *"

successfully links to the Details page but does ***not*** show the answer in the snippet. On the positive side, independently of whether or not the answer is captured in this convenient form, an additional bonus of this option is that, since the query is handled by Google, several other relevant pages from a variety of sites may be brought to the user's attention.

On the other hand, we also experimented with direct searches via Google, using the title ("Geography of Northern Region") or different lists of the keywords contained in the html pages. More often than not, either our Top or Details page, sometimes even both pages, were listed among the results – in this case again the rationale governing the search engine's behaviour is still not clear to us.

[1] see for instance http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html

5. Conclusions

The experiments conducted with the current version of the **WRay1** prototype suggest that our strategy to publish deep Web data is effective. Much work must still be done with respect to a number of topics, such as:

- expressing all entity, attribute and relationship names, and consequently the names of the relational views and of their columns, according to a standard general or domain-oriented vocabulary (e.g. Dublin Core[2])
- introducing criteria to choose keywords in order to maximize both their relevance to users performing queries and the chance of being indexed by search engines
- improving the overall structure of the html tables, particularly to consider the partitioning of the Details page into a set or even a multi-level hierarchy of pages
- enriching the Details page (or pages) with links to additional multimedia information on certain items, including maps, statistics, images, videos, Wikipedia pages, etc.
- investigating tactics to elicit better answers from Google, and doing comparisons with other search engines
- scaling-up the experiments to real practical applications, so as to handle large repositories of data kept by official agencies and/or well-reputed companies
- further automating the *W-Ray* method, noting that only the tasks described at section 3 are undertaken by the current version of the Prolog program

References

Main reference

[0] PICINNINI, H.; LEMOS, M.; CASANOVA, M. A. and FURTADO, A. L. 2010. W-Ray: A Strategy to Publish Deep Web Geographic Data. In *Proc. 4th International Workshop on Semantic and Conceptual Issues in GIS (SeCoGIS 2010)*, Vancouver, Canada - to appear.

Other references (taken from the main reference paper)

[1] BERGMAN, M. K. 2001. The Deep Web: Surfacing Hidden Value. *J. Electr. Pub.* 7(1).

[2] BIZER, C. and CYGANIAK, R., 2006. D2R Server – Publishing Relational Databases on the Web as SPARQL Endpoints. In *Proc. 15th Int'l. WWW Conf.,* Edinburgh, Scotland.

[3] CALDWELL, B.; COOPER, M.; REID, L.G. and VANDERHEIDEN, G. 2008. Web Content Accessibility Guidelines (WCAG) 2.0. In *W3C Recommendation*.

[4] CALLAN J. 2000. Distributed information retrieval. In *Advances in Information Retrieval*, Eds. Springer, US, 127-150.

[5] COSTA L. 2005. Esfinge - Resposta a perguntas usando a Rede. In *Proc. Conf. Ibero-Americana IADIS WWW/Internet*, Lisboa, Portugal.

[6] ERLING, O. and MIKHAILOV, I. 2007. RDF support in the virtuoso DBMS. In *Proc. 1st Conference on Social Semantic Web*, Leipzig, Germany, Vol. 113 of LNI, pp. 59–68.

[7] FLIEDL G.; KOP C. and VÖHRINGER J. 2010. Guideline based evaluation and verbalization of OWL class and property labels. *Data & Knowledge Eng.* 69(4), pp. 331-342.

[8] FUCHS N. E.; KALJURAND K. and KUHN T. 2008. Attempto Controlled English for Knowledge Representation. In *Reasoning Web 2008*, LNCS 5224, pp. 104-124.

[9] GOOGLE. 2008. In *Google's Search Engine Optimization Starter Guide*, Version 1.1.

[10] Alexandria Digital Library, 2004. Guide to the ADL Gazetteer Content Standard, v. 3.2

[11] HOLLINK, L.; SCHREIBER, G.; WIELEMAKER, J. and WIELINGA, B. 2003. Semantic Annotation of Image Collections. *In Proc. Knowledge Markup and Semantic Annotation Workshop*, Sanibel, Florida, USA.

[12] ISO 19115:2003, Geographic Information – Metadata.

[2] http://www.dublincore.org/

[13] KALYANPUR A.; HALASCHEK-WIENER C.; KOLOVSKI V. and HENDLER J. 2005. Effective NL Paraphrasing of Ontologies on the Semantic Web. In *Workshop on End-User Semantic Web Interaction, 4th Int. Semantic Web conference*, Galway, Ireland.
[14] LEME L. A. P. P.; BRAUNER D. F.; CASANOVA M. A. and BREITMAN K. 2007. A Software Architecture for Automated Geographic Metadata Annotation Generation. In *Proc. XXII Simpósio Brasileiro De Banco De Dados, SBBD,* João Pessoa, Brazil.
[15] MADHAVAN J.; AFANASIEV L.; ANTOVA L. and HALEVY A. 2009. Harnessing the Deep Web: Present and Future. In *Proc. 4th Biennial Conf. on Innovative Data Systems Research (CIDR)*, Asilomar, California, USA.
[16] MADHAVAN, J.; KO, D.; KOT, L.; GANAPATHY, V.; RASMUSSEN, A. and HALEVY, A. 2008. Google's Deep-Web Crawl. In *Proc. VLDB* 1(2), pp. 1241–1252.
[17] MapServer. http://mapserver.org/about.html#about
[18] MENG W.; YU C.T. and LIU K.L. 2002. Building efficient and effective metasearch engines. *ACM Computing. Survey,* v. 34, n.1, pp. 48-89.
[19] PRANINSKAS, J. 1975. *Rapid review of English grammar*. Prentice-Hall, NJ, USA.
[20] RAGHAVAN S. and GARCIA-MOLINA H. 2001. Crawling the HiddenWeb. In *Proc. VLDB*, pp. 129-138.
[21] RAJARAMAN A. 2009. Kosmix: HighPerformance Topic Exploration using the Deep Web. In *Proc. VLDB, Lyon, France*.
[22] SORRENTINO S.; BERGAMASCHI S.; GAWINECKI M. and PO L. 2009. Schema Normalization for Improving Schema Matching. In *Proceedings of the 28th International Conference on Conceptual Modeling- ER*, Gramado, Brazil, LNCS 5829, pp. 280-293.
[23] ZHENG, Z. 2002. AnswerBus question answering system. In *Proc. 2nd International Conference on Human Language,* San Diego, California, pp. 399–404.