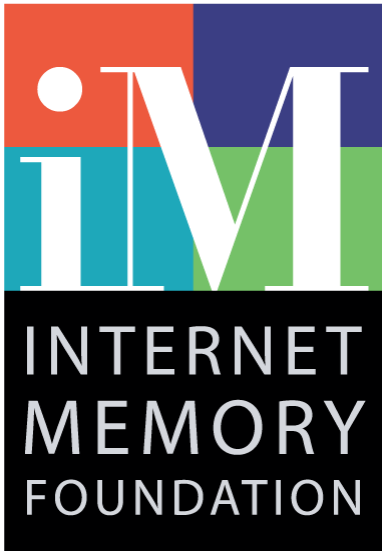
Intelligent Crawling of Web Applications for Web Archiving

{muhammad.faheem, pierre.senellart}@telecom-paristech.fr

Télécom ParisTech (http://dbweb.enst.fr)

WWW PhD symposium. April 20, 2012

Web Archiving



Archiving the Social Web

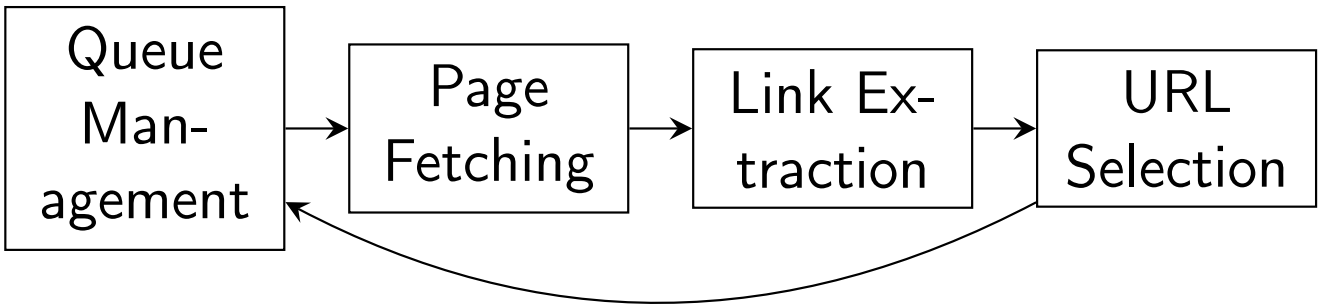
Archiving the Social Web

- Traditional crawling approach crawls the web sites independently of the nature of the site and its content management system.
- Goal: Smart archiving of the Social Web;
  1. Intelligent Crawling
  2. Index Web objects

Agenda

Traditional Crawling Approach

- A traditional Web crawler (such as Heritrix) crawls the Web in a conceptually very simple way.



- This approach does not take into account the nature of Web application.

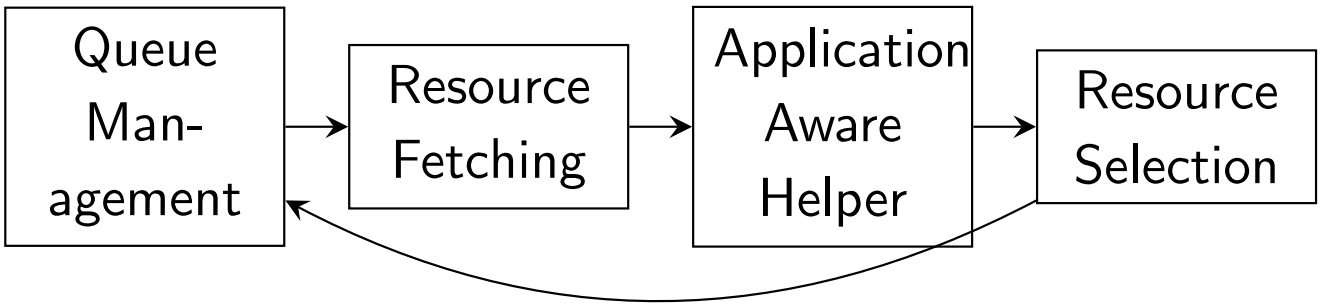
Introduction to Application-Aware Helper

- Different crawling techniques for different social Web sites.
- Detect the type of Web application, kind of Web pages inside this Web application and decide crawling actions accordingly.
- Our approach does not have the same purpose as *focused* crawling.

  Focused Crawling: crawling based on a Topic.
  Application-Aware Helper: crawling optimized for a particular Web Application.

Introduction to Application-Aware Helper

- Extended architecture



- To be implemented in 2 Web crawlers: Internet Memory Foundation crawler, and into Heritrix.

Knowledge base of Web applications

- Knowledge base of Web applications: describes how to crawl a Web site in an intelligent manner.
- Hierarchy: from general categorizations to specific instances (Web sites) of this Web application.
  1. categorizes the web applications.
  2. specifies the detection rules.
  3. describes the specific crawling actions.

Knowledge base of Web applications

- Different crawling actions for different kinds of Web pages under a specific Web application.
- Declarative, XML-based format.

Example of the knowledge base

```
<knowledgebase>
  <cms  name="vBulletin" type="webforum">
   <detection-rules>
     <xpath-expression>
      //script/@src[contains(.,'vbulletin_global.js')]
     </xpath-expression>
   </detection-rules>
   <page-level-cat>
     <list-of-forum>
        <detection-rules>
           <xpath-expression type="1">
            //a[@class="forum"]/@href
           </xpath-expression>
           <xpath-expression type="2">
             //h2[@class="forumtitle"]/a/@href
           </xpath-expression>
        </detection-rules>
```

Example of the knowledge base

```
        <crawling-action>
           <action id="1">
              //a.forum/@href
            </action>
            <action id="2">
              //td.forumtitle/div/a/@href
            </action>
         </crawling-action>
      </list-of-forum>
      <list-of-thread>
            .
      </list-of-thread>
      <thread>
            .
      </thread>
</knowledgebase>
```

Web application detection Module

- One main challenge in intelligent crawling and content extraction is to identify the Web application and then perform the best crawling strategy accordingly.
- Detecting Web application using:
  1. URL patterns,
  2. HTTP metadata,
  3. textual content,
  4. XPath patterns, etc.
- For instance the vBulletin Web forum content management system, that can be identified by searching for a reference to a `vbulletin_global.js` JavaScript script by using a simple `//script/@src` XPath expression.

Crawling and extraction

- **Next stage:** determining the corresponding crawling actions.
- **Crawling action:** not just a list of URLs; can be any action that uses, REST API, complicated interaction with AJAX-based application, and extracts semantic Web objects.

Crawling and extraction

- More specifically, crawling actions are of two kinds:

  Navigation actions: to navigate to another Web page or Web resources.

  Extraction actions: to extract individual semantic objects from Web pages (e.g., timestamp, the blog post, the comments).

Crawling and extraction

- We similarly want a declarative language for describing all crawling actions.
- We therefore need a navigation and extraction language to access data from the deep Web as well as regular URLs.
- We will use OXPath that is an extension of XPath, with added facilities for interacting with Web applications and extracting relevant data.
- It allows the simulation of user actions to interact with scripted multipage interfaces of the Web application.

Crawling and extraction

- As an example, a simple OXPath action that can be performed on the vBulletin example is

```
//a[@class#='posttitle']/@href/{click/}
  /descendant::a[@class#='postbody']
        :<post=string(.)>
```

Initial Results

- Prototype of the application-aware helper implemented, with recognition for a couple of Web applications.
- System evaluated against the vBulletin application, on a variety of Web sites using this content management system.
- For now, the system is able to detect the type of the Web application, the level in the Web application, and executes the corresponding crawling actions.

Initial Results

- Recently, we have integrated the YFilter system (a NFA based filtering system) for efficient indexing of detection patterns, in order to quickly find the relevant Web applications.

Future Work

- Using XPath 1.0 expressions for detection patterns faces some expressiveness limitations, therefore we have the option of switching to XPath 2.0 expressions or to add extension functions for this purpose.
- Automatic, unsupervised, learning of new Web applications (by inference of common patterns), and adaptation to slight changes in the templates that render the wrappers unusable.

Future Work

- Crawler and external program (OXPath evaluator, API crawler, etc.) integration.

Merci