



Adaptive Web Crawling through Structure-Based Link Classification

Muhammad Faheem

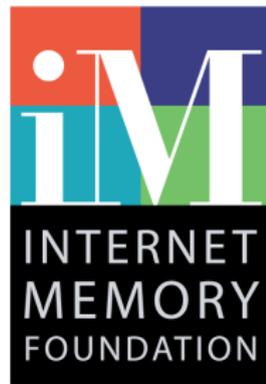


Pierre Senellart





Web archiving





The CMS-Based Web

40–50% of Web content may rely on template-based **content management systems** (CMSs)



The CMS-Based Web

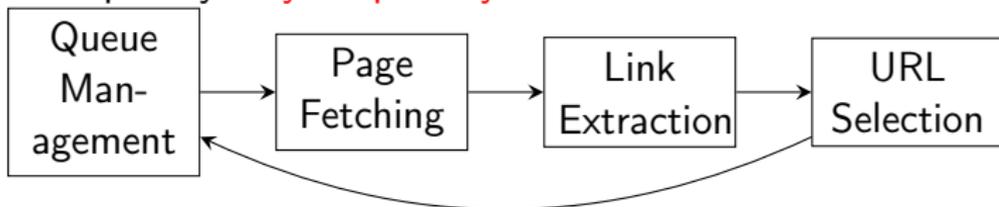
40–50% of Web content may rely on template-based **content management systems** (CMSs)





Traditional crawling approach

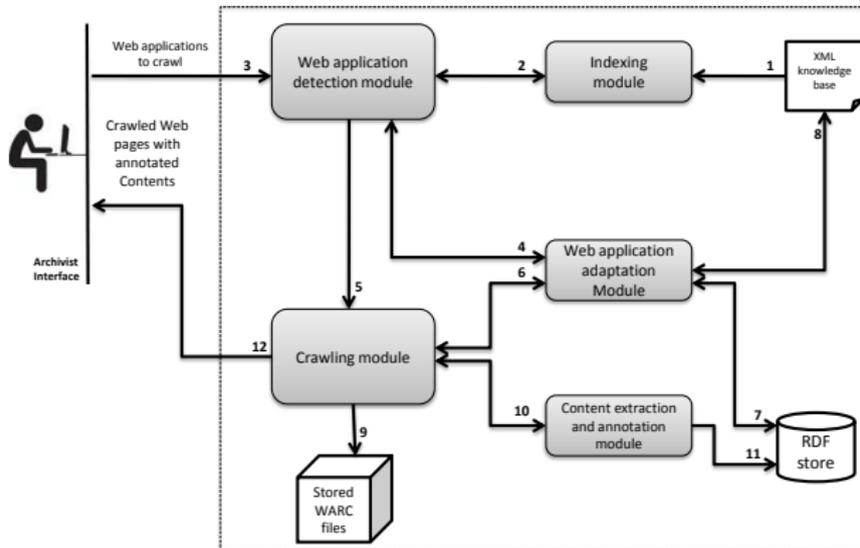
- A traditional Web crawler (such as **Heritrix**) crawls the Web in a conceptually **very simple way**.



- This approach does not take into account the **nature** of the Web site crawled, its **structure**, its **template**.
- Very **inefficient!** Most Web sites have lots of **redundancies**: several URLs host the same content, different ways to navigate within a Web site to the same content, etc.

Application-Aware Helper (AAH)

[ICWE 2013, CIKM 2013]



What is crawled depends on the CMS the Web site is based on, but relies on a **hand-written knowledge base of existing CMSs**



- Fully automated system for Web crawling
- Crawls the content of template-based Web sites with fewer requests than traditional Web crawlers
- Automatically identifies interesting navigation patterns in a Web site
- Adapts to the structure of new Web sites, based on unknown CMSs



Outline

Introduction

Adaptive Crawler Bot for Data Extraction (ACEBot)

Experiments

Conclusion

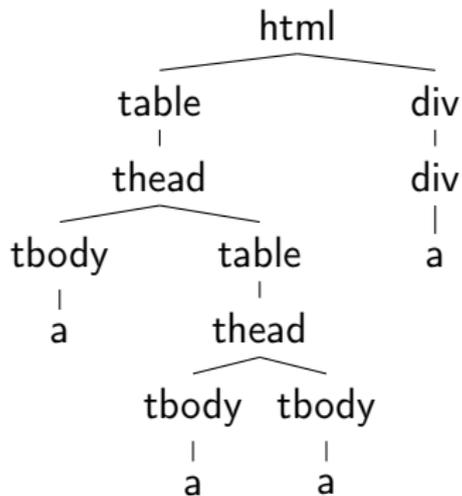


Structure-driven crawler

- ACEBot (**A**daptive **C**rawler **B**ot for data **E**xtraction) is a fully automatic, structure-driven crawler.
- ACEBot **utilizes** the inner structure of the Web pages and guides the crawling process based on the importance of their content.
- ACEBot has two phases:
 - Offline phase:** constructs a dynamic site map (limiting the number of URLs retrieved), learns the best traversal strategy based on importance of navigation patterns (selecting those leading to valuable content).
 - Online phase:** performs **massive** crawling following the chosen navigation patterns.



Simple example: structure of a Web page

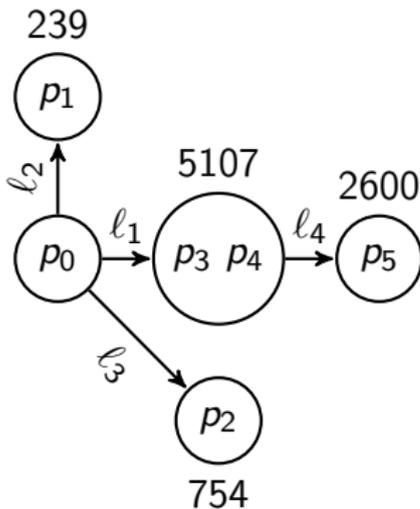
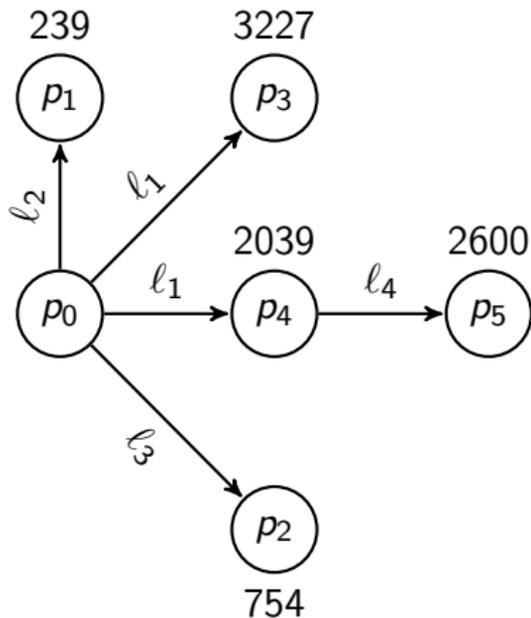


l_1 html-table-thead-table-thead-tbody-a

l_2 html-div-div-a

l_3 html-table-thead-tbody-a

Simple example: structure of a Web site





Objective

- Determine the **navigation patterns** on a Web site that lead to most content items
- **Content items**: can be anything, e.g., **k -grams** (succession of k words)
- **Score of a navigation pattern**:
$$\frac{\text{number of distinct items}}{\text{number of requests}}$$

Proposition

*Finding a set of navigation patterns that optimizes the score is **coNP-hard**.*

⇒ **Greedy** approach, until sufficiently many content items have been retrieved



Objective

- Determine the **navigation patterns** on a Web site that lead to most content items
- **Content items**: can be anything, e.g., **k-grams** (succession of k words)
- **Score of a navigation pattern**:
$$\frac{\text{number of distinct items}}{\text{number of requests}}$$

Proposition

*Finding a set of navigation patterns that optimizes the score is **coNP-hard**.*

⇒ **Greedy** approach, until sufficiently many content items have been retrieved



Objective

- Determine the **navigation patterns** on a Web site that lead to most content items
- **Content items**: can be anything, e.g., **k-grams** (succession of k words)
- **Score of a navigation pattern**:
$$\frac{\text{number of distinct items}}{\text{number of requests}}$$

Proposition

*Finding a set of navigation patterns that optimizes the score is **coNP-hard**.*

⇒ **Greedy** approach, until sufficiently many content items have been retrieved



Language of navigation patterns

Non-disjunctive **regular expressions** over link paths, expressed as **XPath expressions** (language for Web navigation)

$\langle \text{expr} \rangle ::= \text{"doc" " (" \langle \text{url} \rangle \text{"} (\langle \text{estep} \rangle)^+$

$\langle \text{estep} \rangle ::= \langle \text{step} \rangle \mid \langle \text{kleene} \rangle$

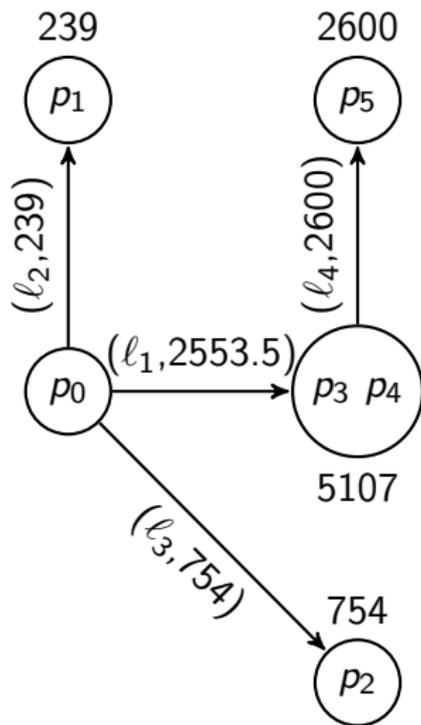
$\langle \text{step} \rangle ::= \text{" / " " (" \langle \text{action} \rangle \text{"} "$

$\langle \text{kleene} \rangle ::= \text{" / " " (" \langle \text{action} \rangle \text{"} (\text{" * " } \mid \langle \text{number} \rangle)$

$\langle \text{action} \rangle ::= (\text{" / " } \langle \text{nodetest} \rangle)^+ \text{" \{ click / \} "$

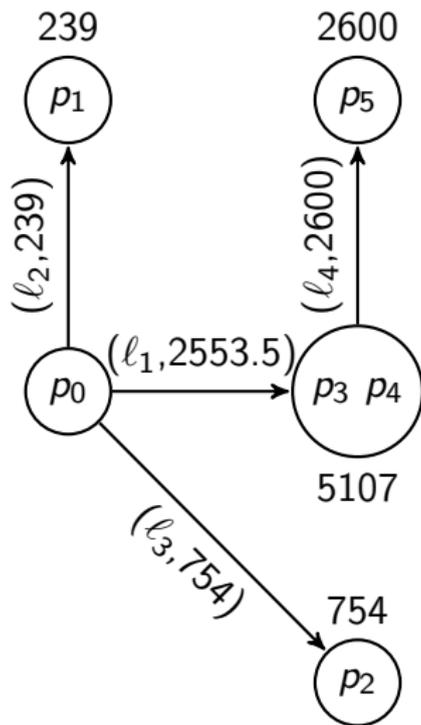
$\langle \text{nodetest} \rangle ::= \text{tag} \mid \text{" @ " tag}$

Simple example: best navigation pattern



NP	total 2-grams	distinct 2-grams	score
l_1	5266	5107	2553.5
l_1, l_4	7866	7214	2404.7
l_3	754	754	754
l_2	239	239	239

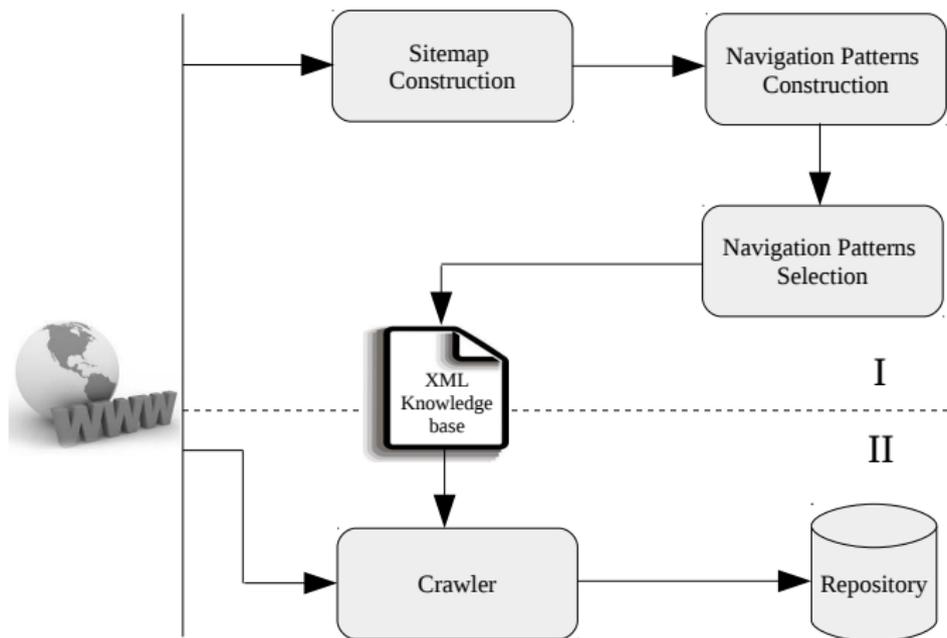
Simple example: best navigation pattern



NP	total 2-grams	distinct 2-grams	score
l_1	5266	5107	2553.5
l_1, l_4	7866	7214	2404.7
l_3	754	754	754
l_2	239	239	239



ACEBot architecture





Outline

Introduction

Adaptive Crawler Bot for Data Extraction (ACEBot)

Experiments

Conclusion



Experiment setup

- The experiments are performed with **ACEBot**, **AHH** and **GNU wget**
- Crawled 50 Web sites (totaling nearly 2 million Web pages) with both **ACEBot** and **wget**
- Crawled 10 Web sites (totaling nearly 0.5 million Web pages) with both **ACEBot** and **AAH**
- Dynamic site map has been constructed with randomly chosen **3000** Web pages

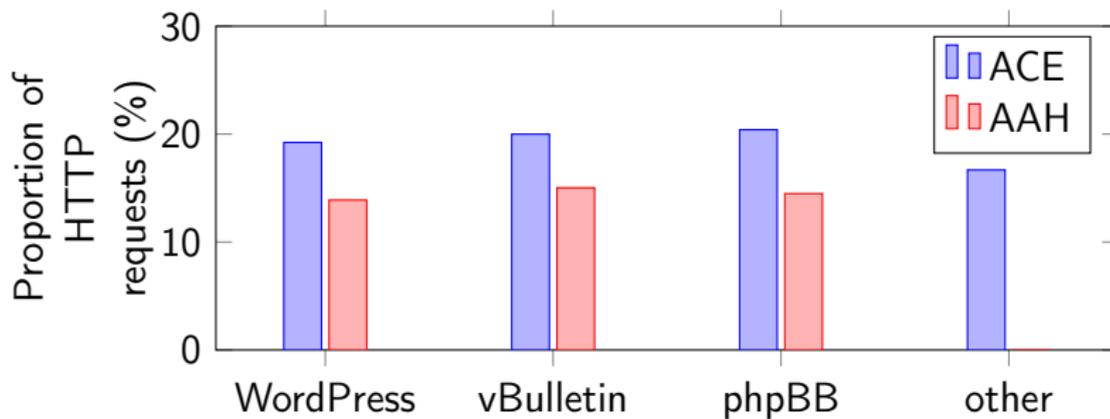


Performance metrics

- The number of HTTP requests made by both systems vs the amount of useful content retrieved
- Coverage of useful content is calculated by comparing the proportion of **2-grams** in the crawl result of both systems for every WA and by counting the number of **external links**

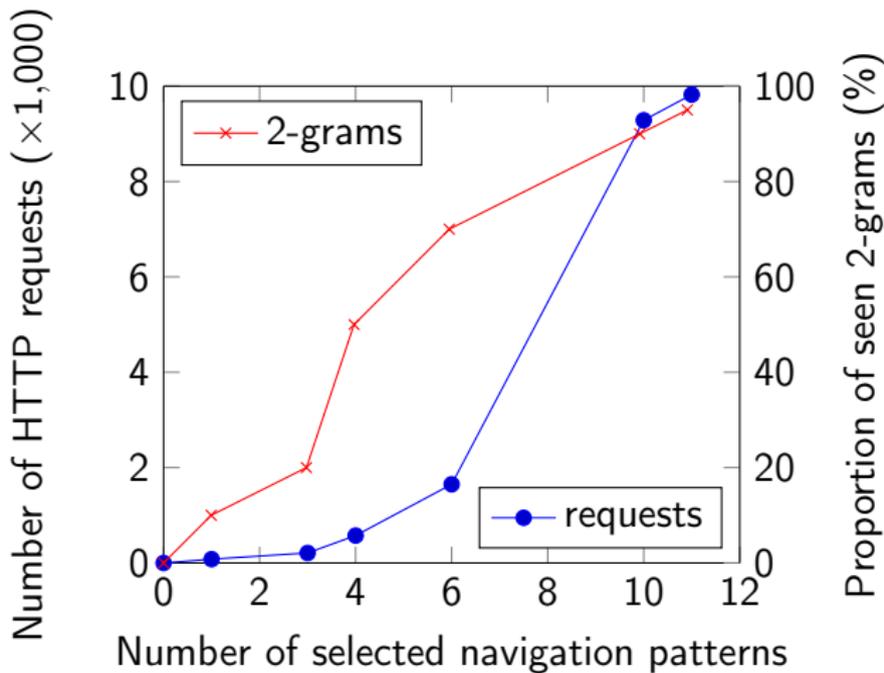


Crawl efficiency



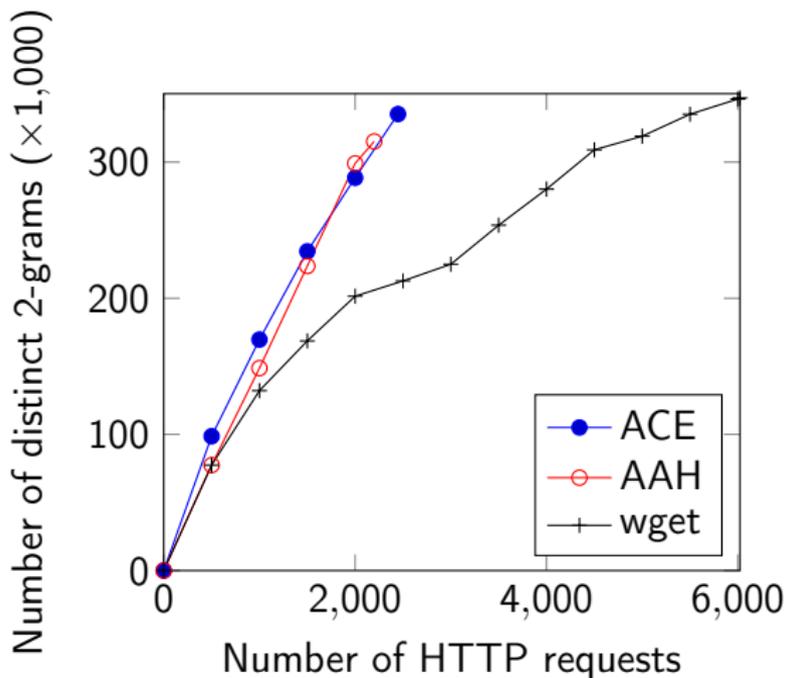


Navigation pattern efficiency



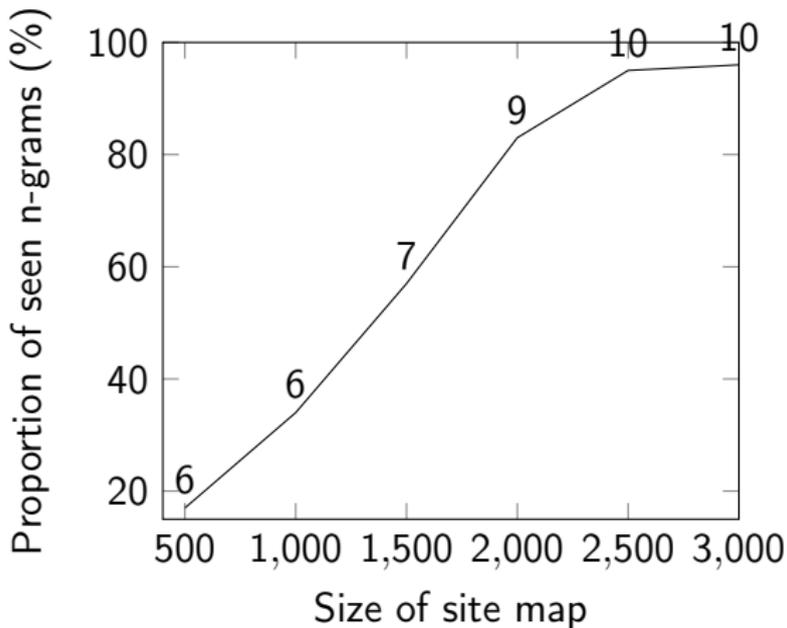


Crawling a particular Web site





Impact of the sitemap size



(Labels are the number of navigation patterns learned.)



Outline

Introduction

Adaptive Crawler Bot for Data Extraction (ACEBot)

Experiments

Conclusion



Conclusion

In brief:

- It is possible to retrieve the significant content of a Web site **much more efficiently** than by crawling it whole
- **Comparable results** between fully automatic approach and approach relying on hand-written knowledge base
- **Sample, learn, apply to whole Web site**

Main open direction:

- More adaptive selection of sitemap size – adapt to the size of the Web site



Conclusion

In brief:

- It is possible to retrieve the significant content of a Web site **much more efficiently** than by crawling it whole
- **Comparable results** between fully automatic approach and approach relying on hand-written knowledge base
- **Sample, learn, apply to whole Web site**

Main open direction:

- More adaptive selection of sitemap size – adapt to the size of the Web site

Merci !