Examples of web crawler architectures

Sarah Koenig

TU Dortmund

22.11.2007
Some remarks before I begin...

Two helpful websites:

- www.m-w.com
  listen to the pronunciation of difficult words (e.g. frontier, hierarchically, archiv ... )

- www.leo.org
  german – english
Introduction

1 Introduction
- What is a web crawler?
- The basic idea of crawling
- The original Google crawler
Introduction

1 Introduction
- What is a web crawler?
- The basic idea of crawling
- The original Google crawler

2 Mercator – a high performance web crawler
- Mercator – general layout
- Mercator as a distributed system
  - Organizing the processes
  - Connecting the processes
- Mercator – the URL frontier in detail
  - Overview
  - URL frontier front-end
  - URL frontier back-end
1 Introduction
   - What is a web crawler?
   - The basic idea of crawling
   - The original Google crawler

2 Mercator – a high performance web crawler
   - Mercator – general layout
   - Mercator as a distributed system
     - Organizing the processes
     - Connecting the processes
   - Mercator – the URL frontier in detail
     - Overview
     - URL frontier front-end
     - URL frontier back-end
What is a web crawler?

- Crawler, robot, spider, worm, walker, wanderer
What is a web crawler?

- Crawler, robot, spider, worm, walker, wanderer
- “A machine, which runs through the web where it downloads and looks at every page.”
What applications use web crawlers?

Web crawlers are used for

- Indexing pages for search engines
- Archiving the web (e.g., The Internet Archive www.archive.org)
- Analysing the web
- Collecting email addresses for spammers
What applications use web crawlers?

Web crawlers are used for

- indexing pages for search engines
- archiving the web (e.g. The Internet Archive www.archive.org)
- analysing the web
- ...

Examples of web crawler architectures
What applications use web crawlers?

Web crawlers are used for

- indexing pages for search engines
- archiving the web (e.g. The Internet Archive www.archive.org)
- analysing the web
- ...
- collecting email addresses for spammers
## Overview

1. **Introduction**
   - What is a web crawler?
   - The basic idea of crawling
   - The original Google crawler

2. **Mercator – a high performance web crawler**
   - Mercator – general layout
   - Mercator as a distributed system
     - Organizing the processes
     - Connecting the processes
   - Mercator – the URL frontier in detail
     - Overview
     - URL frontier front-end
     - URL frontier back-end
Perform a Breadth-first Search, starting with one document:

1. find all URLs in the document
2. follow each link and download the document ...

But this is not efficient!
Web crawler requirements

What are the requirements for web crawlers?
Web crawler requirements

What are the requirements for web crawlers?

- distributed
- scalable
- portable
- high performance
  (e.g. 50 million documents / 17 days with 4 Compaq server)
- continuous
- extensible
- polite
Introduction

Mercator – a high performance web crawler

What is a web crawler?

The basic idea of crawling

The original Google crawler

Overview

Mercator – a high performance web crawler

Mercator – general layout

Mercator as a distributed system
  - Organizing the processes
  - Connecting the processes

Mercator – the URL frontier in detail
  - Overview
  - URL frontier front-end
  - URL frontier back-end
The original Google crawler

- consists of 5 different components
  1. URL server process – provides URLs
  2. 4 crawler processes, each running single threaded on a different machine
  3. StoreServer process – stores the downloaded file to disk
  4. Indexer process – extracts URLs
  5. URL resolver – derelativizes and stores URLs

- ran on 8 machines
The original Google crawler
The original Google crawler

Introduction
Mercator – a high performance web crawler

What is a web crawler?
The basic idea of crawling
The original Google crawler

Examples of web crawler architectures
The original Google crawler
The original Google crawler

- URL server process
- Crawler processes
- Store server process
- Indexer process
- URL resolver process
- URLs
Introduction
Mercator – a high performance web crawler

The original Google crawler

The basic idea of crawling
URL server process
1 crawler process per machine
URLs

Examples of web crawler architectures
The original Google crawler
The original Google crawler
The original Google crawler

Diagram:
- URL server process
- Crawler processes
- Store server process
- Indexer process
- URL resolver process
- Derelativize the links
- URLs
The original Google crawler

- URL server process
  - processes
  - store server process
  - indexer process
  - URL resolver process

Examples of web crawler architectures
The original Google crawler

- URL server
- server process
- crawler processes
- store server process
- indexer process
- URL resolver process
- URLs
Overview

1. Introduction
   - What is a web crawler?
   - The basic idea of crawling
   - The original Google crawler

2. Mercator – a high performance web crawler
   - Mercator – general layout
   - Mercator as a distributed system
     - Organizing the processes
     - Connecting the processes
   - Mercator – the URL frontier in detail
     - Overview
     - URL frontier front-end
     - URL frontier back-end

Examples of web crawler architectures
Mercator was designed to meet high requirements:

- distributed
- scalable
- portable
- high performance
- continuous
- extensible
- polite (weak / strong politeness)
To improve the scalability, Mercator consists of identical processes.

Each process has multiple worker threads.
Mercator – layout of a worker thread
Mercator – layout of a worker thread
Mercator – layout of a worker thread

- Frontier = “unknown land beyond a border”
- Protocol module according to scheme (http://... ftp://...)

---

Sarah Koenig
Examples of web crawler architectures
DNS resolving is slow (and synchronized in Java) ⇒ custom DNS resolver with local DNS server
• RIS (Rewind Input Stream) is a stream which can be read more than once.
Mercator – layout of a worker thread

- Compute and compare fingerprint of the document
Mercator – layout of a worker thread

- Extracts and derelativizes links
Mercator – layout of a worker thread

- Other processing modules can be plugged in.
Mercator – layout of a worker thread

- User-supplied URL filter (e.g. to exclude crawler traps)
DUE (Duplicate URL Eliminator) computes and compares fingerprint of the URL.
Mercator – layout of a worker thread
Overview

1. Introduction
   - What is a web crawler?
   - The basic idea of crawling
   - The original Google crawler

2. Mercator – a high performance web crawler
   - Mercator – general layout
   - Mercator as a distributed system
     - Organizing the processes
     - Connecting the processes
   - Mercator – the URL frontier in detail
     - Overview
     - URL frontier front-end
     - URL frontier back-end
Organizing the processes
Organizing the processes

- analog to insects which live in colonies
Organizing the processes

- Analog to a colony of insects
Organizing the processes

- background thread creates checkpoints, logs summary statistics...
But how do we prevent processes from downloading the same document?
Organizing the processes

- Each process is responsible for a set of hosts
Organizing the processes

- URLs which do not belong to the own “realm” are forwarded to the corresponding process
Organizing the processes

- URLs which do not belong to the own “realm” are forwarded to the corresponding process
Organizing the processes

- URLs which do not belong to the own “realm” are forwarded to the corresponding process
Organizing the processes

- URLs which do not belong to the own “realm” are forwarded to the corresponding process
Introduction
Mercator – a high performance web crawler

Mercator – general layout
Mercator as a distributed system
Mercator – the URL frontier in detail

Connecting the processes

Example of web crawler architectures
Connecting the processes

![Diagram showing the process flow]

- HTTP
- Protocol Modules
- RIS
- Link Extractor
- Processing Modules
- Content Seen?
- Doc FPs
- URL Filter
- DUE
- URL Frontier
Connecting the processes
Connecting the processes

Introducing Mercator - a high performance web crawler

Mercator - general layout
Mercator as a distributed system
Mercator - the URL frontier in detail

Example of web crawler architectures
Connecting the processes
Connecting the processes

![Diagram of Mercator web crawler processes](image-url)
A crawling hive

Sarah Koenig

Examples of web crawler architectures
Overview

1. Introduction
   - What is a web crawler?
   - The basic idea of crawling
   - The original Google crawler

2. Mercator – a high performance web crawler
   - Mercator – general layout
   - Mercator as a distributed system
     - Organizing the processes
     - Connecting the processes
   - Mercator – the URL frontier in detail
     - Overview
     - URL frontier front-end
     - URL frontier back-end
Overview

Mercator – a high performance web crawler

Mercator – general layout
Mercator as a distributed system
Mercator – the URL frontier in detail

Sarah Koenig
Examples of web crawler architectures
Overview

- Consists of a front-end and a back-end
- The front-end manages priorities
- The back-end implements politeness
The front-end:
The front-end:

Front-end FIFO queues (one per priority level)

Random queue chooser with bias to high-priority queues
Front-end

![Diagram of Mercator's front-end components]

- HTTP
- Protocol Modules
- RIS
- Link Extractor
- Processing Modules
- URL Filter
- DUE
- URL Frontier

Content Seen?

Doc FP
Front-end

The front-end:

1. URL is submitted to the URL frontier
Front-end

The front-end:

1. URL is submitted to the URL frontier
2. Prioritizer computes the priorities

Front-end FIFO queues (one per priority level)
Random queue chooser with bias to high-priority queues
The front-end:

1. URL is submitted to the URL frontier
2. Prioritizer computes the priorities
3. URL is sorted into a FIFO
4. Front-end FIFO queues (one per priority level)
5. Random queue chooser with bias to high-priority queues
The front-end:

1. URL is submitted to the URL frontier
2. Prioritizer computes the priorities
3. URL is sorted into a FIFO
4. when the back-end requests a URL, a queue is selected.

Queues with a high priority level are selected with higher probability.
The back-end

![Diagram of back-end queue router and FIFO queues]

- **Back-end queue router**
- **FIFO queues** (many more than worker threads)
- **Host-to-queue table**
  - A → 3
  - C → 1
  - F → n
  - X → 2
- **Priority queue** (e.g., heap)
  - 2
  - n
  - 1
  - 3
The back-end consists of

- nonempty FIFOs – one per host
The back-end consists of:

- nonempty FIFOs – one per host
- a Host-to-queue table
The back-end consists of

- nonempty FIFOs – one per host
- a Host-to-queue table
- a queue for the waiting time

waiting time = downloading time \times 10
⇒ strong politeness
When one of the FIFOs becomes empty, the back-end queue router requests a URL from the front-end.
Filling the back-end

Back-end queue router checks if there is already a queue for the host.
Filling the back-end

True ⇒ submit URL to the queue and request another URL from the front-end

False ⇒ submit the URL to the empty queue
Selecting a URL

- HTTP
- RIS
- Link Extractor
- URL Filter
- DUE
- URL Frontier

Content Seen?
- Doc FPs

Protocol Modules
- Processing Modules
Selecting a URL

The back-end queue selector chooses a queue from which a URL will be taken. By this it decides from which host the crawler will download next.
Selecting a URL

The host is selected respecting the waiting times. Waiting times are organized in a priority queue.