Full Text Search of Web Archive Collections

IWAW2005

Michael Stack

Internet Archive

stack@archive.org
Internet Archive

- IA is a 501(c)(3) non-profit
- Mission is to build a public Internet digital library.
  - Movies, Books, Web, etc.
  - “Universal Access to All Human Knowledge”
- This talk is about developing access to Web Collections
IA Web Collection

- IA Web Archive Collection currently:
  - 600TB of data
  - 60 Billion URLs (Google 8 Billion).
- Bulk of Web Collection donated by Alexa Internet
  - Private company, crawling since 1996.
- 2-month rolling snapshots
  - Recent: 4.7 billion URIs, 180 million websites, 44TB.
- Crawling software
  - Sophisticated
  - Weighted towards popular sites
  - Proprietary: We only receive the data
IA Web Collection Access

• Currently, only public access to Web Collection is via the...
Public Access Limitation

• But Wayback Machine has major shortcomings:
  – You must know the exact URL of the page you want to browse beforehand.
    – Can use URLs from published sources.
    – Can use search engines to first find URLs.
  • Does not allow for serendipitous exploration of past.
  • Particularly frustrates users accustomed to Google-style full text query box.
    – IA Wayback Machine cannot be distributed.

• Need full text search of Web Archive Collections!
Not Just IA...

- Others also need to be able to full text search Web Archive Collections. E.g.,
  - Libraries developing own WACs.
  - Users of HIRUX the IA's open source, extensible, archival-quality, web crawler.
Searching Web Archive Collections

• Has much in common searching any large collection of Web resources.
  – Parse common types.
    • text/*, application/pdf, application/msword, etc.
  – Return quality search results.
  – Scales, Performant, Cheap, Customizable, etc.

• But also challenges unique to Web Archive Collections (WACs).
What is a Web Archive Collection?

- Web Archive Collections (WACs) are typically an aggregate of multiple, related, focused Web crawls. Examples:
  - The IA has been making a Hurricane Katrina WAC crawling Katrina sites on a 3-day period.
  - The IA ran 34 weekly crawls of sites pertaining to the US 2004 Presidential election.
WAC Attributes

• Key attributes of WACs:
  – Tend to be large. E.g.:
    • 34 weekly US 2004 election crawls ~40 million.
    • 60 Billion URLs of the IA Web collection is a WAC
  – A single URL may appear multiple times.
    • 34 instances of http://www.whitehouse.gov in US 2004 election WAC.
  – No help from Web displaying found pages.
    • Click on search result takes you where? 1999? 2001?
    • A Wayback-like viewer application or Search Engine Cache required – renders pages from past in browser
Nutch as Search platform

• Nutch, lucene.apache.org/nutch, selected as search engine platform on which to develop WAC search.
  – "Nutch is a complete open-source Web search engine package that aims to index the World Wide Web as effectively as commercial search services"
  – Technically, Nutch provides basic search engine capability, is extensible, aims to be cost-effective, and is demonstrated capable of indexing up to 100 million documents.
    • Convincing development story for how to scale up to billions (More on this later).
  – Just as importantly, policy-wise, Nutch project aligns with mission of IA/IIPC:
    • Transparent alternative to commercial web search engines.
      – Signature feature is ability to explain search result rankings.
        » See 09/17 FT excerpt from John Battelle 'The Search'. Horror stories of how the inscrutable algorithm can change a small business's fortune overnight.
Nutch Overview

- Popular open source project.
- Java.
- Builds on Lucene search lib, adding: crawler, a link-graph database, parsers for common types, etc.
- Customizable at parse-time, index-time, and query-time via plugins.
  - Add/Amend query terms.
  - Add/Amend parsers.
Nutch Overview: Indexing 1

- Runs in stepped, batch mode.
  - Should a step fail (machine crash or operator misconfiguration), just redo.

- First step, "segment" the work
  - No single step overpowers as collection grows.
    - In current implementation, not the case (Will revisit later below).
  - Can distribute “segments” across machines.

- Custom DB maintains state.
  - Between steps and across segments.
  - Computes link structure.
Nutch Overview: Indexing 2

• Steps:
  1. Ask Nutch DB to generate URLs to fetch.
  2. Fetch and Parse the downloaded pages.
  3. Update Nutch DB, run analysis on DB content, update segments.
  4. Index parsed text + in-link anchor text (From Nutch DB).
  5. Optionally remove duplicates (By URL + content MD5).
  6. Optionally merge all segment indices.

• In current Nutch (<= v0.7) implementation:
  – Steps 2 and 4 may be distributed/run in parallel.
  – All other steps require either single process exclusive access to Nutch DB or single process exclusive access to all segment data.
  – A step must complete before the next can begin.
Nutch Overview: Querying

• Start the Nutch search Web application.
  – Run multiple to distribute query processing.
    • Distributes by remotely invoking queries against all query cluster participants.
      – Each query cluster participant is responsible for some subset of all “segments”.

• Queries return ranked Google-like results.
  – Support for basic query terms: url, site, etc.
Adapting Nutch to WAC Search

• WAC search needs to support 2 distinct modes of operations.

1. As Google-like search engine.
   • No duplicate URL pollution in results.

2. Allows study of how pages change over time
   • "return all versions of www.whitehouse.gov crawled in 1999 sorted by crawl date".
     - This is what IA Wayback Machine does.
     - IA Wayback Machine cannot do “…and contains terms 'Hilary' and 'Clinton'”

• Also add support for IA WM-like WAC viewer application.
Adapting Nutch: Mode 1

• Nutch fetcher step recast to pull content from a WAC repository rather than from the live Web.
  – WAC content already exists, previously harvested by other means.
    • At IA harvested content is stored in ARC files.
    • ARC-to-segment tool feeds ARCs to Nutch parsers and segment content writers.
      – Adaptation for formats other IA ARC, trivial.

• Upon completion, using Nutch indices purged of exact duplicates, possible to deploy basic WAC search using IA Wayback Machine as WAC viewer.
Adapting Nutch: Mode 2

- Added following to support WAC viewer and wayback-like querying:
  - Added support for explicit IA 14-digit – YYYYDDMMHHSS -- date and date range querying.
    - Replaced Nutch native date support.
  - Added ARC location information to search result: *collection*, *arcname*, and *arcoffset*.
  - Added default parser. Used when no parser match found (e.g. types w/o text). Adds meta-info on each resource.
    - Allows stylesheets, images, etc., to be found by WAC viewer drawing pages from past.
  - Native Nutch modified to support sort on arbitrary fields: *sort*, *reverse*.
  - Native Nutch modified to support removal of duplicates at query time (rather than at index time): *hitsPerDup*, *dedupField*.
  - Native Nutch modified to return results as XML (A9 OpenSearch RSS).

- Upon completion, both modes of operation possible using same non-deduplicated index. Example queries:
  - *hilary clinton site:*www.whitehouse.gov date:1999-2000 sort:date
  - *levees collection:*katrina date:200508-200509 sort:date
Nutchwax

- All Nutch WAC plugin extensions, documentation, and scripts are open source, hosted on Sourceforge under the *Nutchwax -- Nutch with Web Archive eXtensions* – project: http://archive-access/projects/nutch/
Running WAC Search: Indexing Stats

• Indexing Machine Profile
  – Single processor 2.80GHz Pentium 4s with 1GB of RAM and 4x400GB IDE disks running Debian GNU/Linux.
  – Indexing, CPU-bound with light I/O loading.
  – RAM sufficient (no swapping).
  – All source ARC data NFS mounted.
• Only documents of type text/*/ or application/*/ and HTTP status code 200 were indexed.
Indexing Stats: Small Collection

• Collection
  – Three crawls.
  – Indexing steps run in series on one machine using single disk.
  – 206 100MB ARC files, 37.2GB of uncompressed data.
  – 1.07 million documents indexed.

• Indexing
  – 40.3 hours to complete.
    • 1/3rd segmenting/parsing, 1/3rd indexing, 1/3rd all other steps.
  – Merged index size was 3% the size of src.
    • Index plus cleaned-up segments occupied 46% src.
    • Index plus uncleaned segments made up 40% src.
Indexing Stats: Medium Collection

• Collection
  – 1054 ARCs, 147.2GB of uncompressed data.
  – 4.1 million documents indexed.
  – Two machines to do the segmenting step.
  – Subsequent steps all run in series on a single machine using a single disk.

• Indexing
  – 99 hours of processing time
    • Or 86.4 hours of elapsed time because segmenting split.
  – Merged index size was 5.2GB, 4% source.
  – Index plus cleaned-up segment data 13.5% source.
  – Index plus uncleaned segments 22% source.
Observations 1

• Indexing big collections is a long-running manual process.
  – Requires manual intervention at each step moving process along.
  – Attention compounds the more distributed the indexing.
  – An early indexing of 85 million took approx. a week over 4 machines.
    • Steps restarted as disks overfilled.
    • Little science applied so load suboptimally distributed.
    • Synchronizations waiting on laggard processes.
  – Current toolset, vigilant operator, a week of time, 4 to 5 machines with lots of disk, indexing 100 million doc. WACs is practical limit (200 million documents...perhaps...if segments and indices).

• Automated means of efficiently distributing indexing needed!
  – But some indexing steps are currently single process.
  – And as collection grows, so grows central Nutch DB of page and link content. Eventually larger than any available single disk.
Observations 2

- Inclusion of in-link anchor-text indexing improves search result quality.
  - Without, results rich in query terms but *wrong*.

- Distributed Nutch query clustering works well.
  - At least for low rates of access: ¼ query per second.
    - Search access-rates are expected lower for WACs than live-Web search.
  - But caches to speed querying will turn problematic.
    - Nutch (Lucene) query implementation uses one byte per document per field indexed.
    - Additions made to support query-time deduplication and sorting share cache of each search result's document URL. Such a cache of (Java) UTF-16 Java strings gets large fast.

- Collections of a few million plus hosted on single disk/single machine, show distinct lag drawing search results.
  - Distribute segments over disks and machines.

- Robust, performant, (Java) doc-type parsers are sparse.
  - Especially for proprietary types: application/pdf, msword, etc.
  - External call out to XPDF for application/pdf (Still slow relative text/*).
Future 1

• Nutch project moving onto distributed file system.
  – Nutch Distributed File System (NDFS).
    • "...software for storing very large stream-oriented files over a set of commodity computers. Files are replicated across machines for safety, and load is balanced fairly across the machine set"
    • Java implementation of subset of Google File System (GFS).
    • Nutch DB distributed, segments distributed.

• How to evenly distribute work across cluster?
  – MapReduce!
    • "a platform on which to build scalable computing"
    • Another Google innovation.
    • Cast cluster task in MapReduce mold -- think Python map function followed by reduce function -- then the MapReduce platform will manage distribution of task across cluster in fault-tolerant way.
    • Java implementation of Google MapReduce underway in Nutch project (Doug Cutting, Mike Cafarella).
• Viewer applications
  – NWA WERA viewer using Nutchwax engine (October 2005).
  – Open source Wayback Machine.

• IA moving Web collections to Petabox platform.
  – Racks of low power, high storage density, inexpensive, rack-mounted, “shipping container friendly” computers.
    • Open source design.
    • Previously, racks of commodity PCs.
• Future WAC Search development.
  – Immediate: Nutchwax 0.4 release based on Nutch 0.7 coordinated with release of the WERA viewer (Lots of bug fixes, October 2005).
  – Beyond: Scale beyond 400 million limit to WACs of 1 Billion and beyond.
    • Exploitation of Nutch project NDFS/MapReduce platform atop Petabox.

Thank you