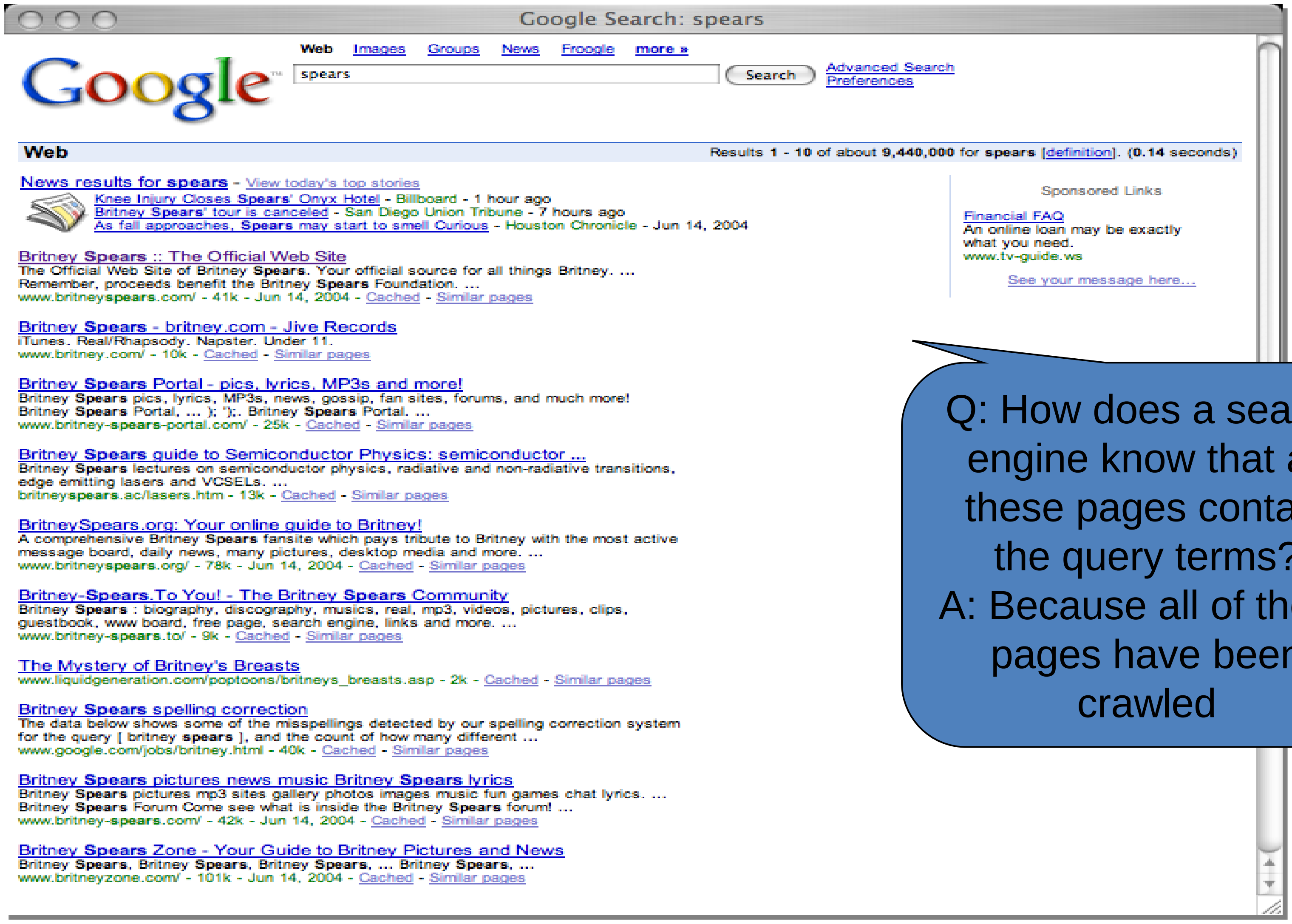
Introduction to Web Crawling

Ramesh Ragala
Assistant Professor (Senior)
VIT Chennai

Web Crawling



Web Results **1** - **10** of about **9,440,000** for **spears** [definition]. (**0.14** seconds)

News results for **spears** - View today's top stories
Knee Injury Closes **Spears**' Onyx Hotel - Billboard - 1 hour ago
Britney **Spears**' tour is canceled - San Diego Union Tribune - 7 hours ago
As fall approaches, **Spears** may start to smell Curious - Houston Chronicle - Jun 14, 2004

Britney **Spears** :: The Official Web Site
The Official Web Site of Britney **Spears**. Your official source for all things Britney. ...
Remember, proceeds benefit the Britney **Spears** Foundation. ...
www.britney**spears**.com/ - 41k - Jun 14, 2004 - Cached - Similar pages

Britney **Spears** - britney.com - Jive Records
iTunes. Real/Rhapsody. Napster. Under 11.
www.britney.com/ - 10k - Cached - Similar pages

Britney **Spears** Portal - pics, lyrics, MP3s and more!
Britney **Spears** pics, lyrics, MP3s, news, gossip, fan sites, forums, and much more!
Britney **Spears** Portal, ...); ");. Britney **Spears** Portal. ...
www.britney-**spears**-portal.com/ - 25k - Cached - Similar pages

Britney **Spears** guide to Semiconductor Physics: semiconductor ...
Britney **Spears** lectures on semiconductor physics, radiative and non-radiative transitions,
edge emitting lasers and VCSELs. ...
britney**spears**.ac/lasers.htm - 13k - Cached - Similar pages

BritneySpears.org: Your online guide to Britney!
A comprehensive Britney **Spears** fansite which pays tribute to Britney with the most active
message board, daily news, many pictures, desktop media and more. ...
www.britney**spears**.org/ - 78k - Jun 14, 2004 - Cached - Similar pages

Britney-**Spears**.To You! - The Britney **Spears** Community
Britney **Spears** : biography, discography, musics, real, mp3, videos, pictures, clips,
guestbook, www board, free page, search engine, links and more. ...
www.britney-**spears**.to/ - 9k - Cached - Similar pages

The Mystery of Britney's Breasts
www.liquidgeneration.com/poptoons/britneys_breasts.asp - 2k - Cached - Similar pages

Britney **Spears** spelling correction
The data below shows some of the misspellings detected by our spelling correction system
for the query [britney **spears**], and the count of how many different ...
www.google.com/jobs/britney.html - 40k - Cached - Similar pages

Britney **Spears** pictures news music Britney **Spears** lyrics
Britney **Spears** pictures mp3 sites gallery photos images music fun games chat lyrics. ...
Britney **Spears** Forum Come see what is inside the Britney **Spears** forum! ...
www.britney-**spears**.com/ - 42k - Jun 14, 2004 - Cached - Similar pages

Britney **Spears** Zone - Your Guide to Britney Pictures and News
Britney **Spears**, Britney **Spears**, Britney **Spears**, ... Britney **Spears**, ...
www.britneyzone.com/ - 101k - Jun 14, 2004 - Cached - Similar pages

Sponsored Links

Financial FAQ
An online loan may be exactly
what you need.
www.tv-guide.ws

See your message here...

Q: How does a search engine know that all these pages contain the query terms?
A: Because all of those pages have been crawled

Web Crawling

- Web Crawlers are programs that automatically download web pages.
- Web search is the quintessential large-data problem.
- Many Names
  - Crawler
  - Spider
  - Robot (or bot)
  - Web agent
  - Wanderer, worm, …
  - And famous instances: googlebot, scooter, slurp, msnbot, …

Web Crawling

- Web Crawlers are programs that automatically download web pages.
- A crawler can **visit** many sites to collect information that can be analyzed and mined in a central location, either online (as it is downloaded) or off-line (after it is stored).
- Simply, Crawling is the process by which search engines discover updated content on the web, such as new sites or pages, changes to existing sites and dead links.

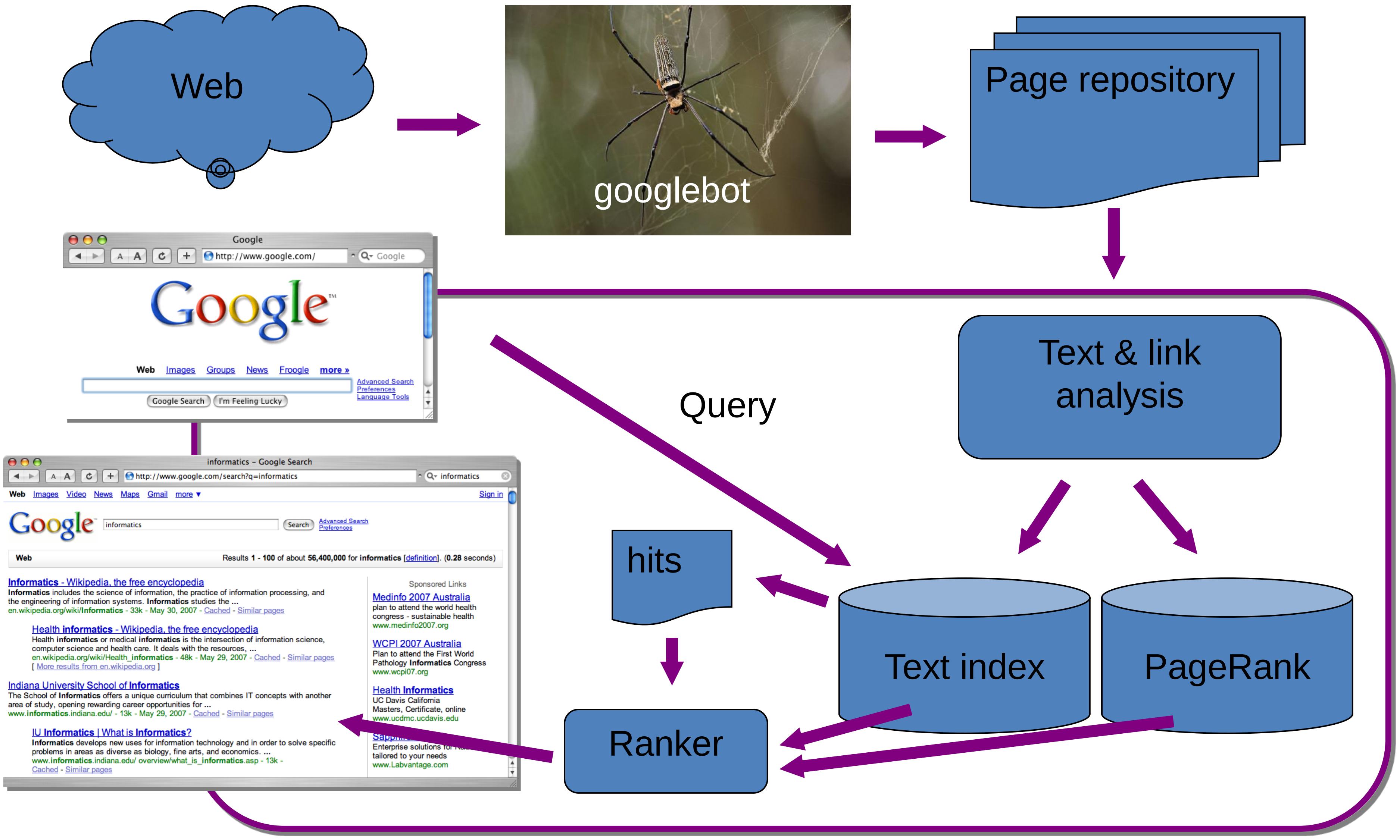
Web Crawling

- Given an information need expressed as a short query consisting of a few terms, the system's task is to retrieve relevant web objects and present them to the user.
- Web objects:
  - web pages
  - PDF documents
  - PowerPoint slides, etc.
- It is difficult to compute exactly, but even a conservative estimate would place the size at several tens of billions of pages, totalling hundreds of terabytes .
- User will expect results in quick manner

Web Crawling

- Applications:
    - Support universal search engines (Google, Yahoo, MSN/Windows Live, Ask, etc.)
    - Vertical (specialized) search engines, e.g. news, shopping, papers, recipes, reviews, etc.
    - Business Intelligence: keep track of potential competitors, partners
    - Monitor Web pages and give notification to users or community
    - Phishing, spamming,etc

A crawler within a search engine



informatics – Google Search

http://www.google.com/search?q=informatics

Web Images Video News Maps Gmail more ▾ Sign in

informatics Search Advanced Search Preferences

Web Results **1** - **100** of about **56,400,000** for **informatics** [definition]. (**0.28** seconds)

Informatics - Wikipedia, the free encyclopedia
Informatics includes the science of information, the practice of information processing, and the engineering of information systems. **Informatics** studies the ...
en.wikipedia.org/wiki/**Informatics** - 33k - May 30, 2007 - Cached - Similar pages

Health **informatics** - Wikipedia, the free encyclopedia
Health **informatics** or medical **informatics** is the intersection of information science, computer science and health care. It deals with the resources, ...
en.wikipedia.org/wiki/Health_**informatics** - 48k - May 29, 2007 - Cached - Similar pages
[More results from en.wikipedia.org]

Indiana University School of **Informatics**
The School of **Informatics** offers a unique curriculum that combines IT concepts with another area of study, opening rewarding career opportunities for ...
www.**informatics**.indiana.edu/ - 13k - May 29, 2007 - Cached - Similar pages

IU **Informatics** | What is **Informatics**?
Informatics develops new uses for information technology and in order to solve specific problems in areas as diverse as biology, fine arts, and economics. ...
www.**informatics**.indiana.edu/ overview/what_is_**informatics**.asp - 13k -
Cached - Similar pages

Sponsored Links

Medinfo 2007 Australia
plan to attend the world health congress - sustainable health
www.medinfo2007.org

WCPI 2007 Australia
Plan to attend the First World Pathology **Informatics** Congress
www.wcpi07.org

Health **Informatics**
UC Davis California
Masters, Certificate, online
www.ucdmc.ucdavis.edu

Sapph
Enterprise solutions for
tailored to your needs
www.Labvantage.com

Web Crawling

- Web Crawlers are used to support search engines.
- Crawlers are main consumers of Internet bandwidth
- Web crawlers collect pages for search engines to build their indexes.
- Nearly all retrieval engines for full-text search today rely on a data structure called **an inverted index**, which given a term provides access to the list of documents that contain the term
- In information retrieval parlance, objects to be retrieved are generically called “documents"

Web Crawling

- Given a user query, the retrieval engine uses the **inverted index** to score documents that contain the query terms with respect to some ranking model, taking into account features such as term matches, term proximity, attributes of the terms in the document as well as the hyperlink structure of the documents.
- The web search problem decomposes into three components:
- 1. gathering web content (crawling).
- 2. construction of the inverted index (indexing)
- 3. ranking documents given a query (retrieval).
- Crawling and indexing share similar characteristics and requirements, but these are very different from retrieval.

Web Crawling

- Gathering web content and building inverted indexes are for the most part offline problems.
- Indexing is usually a batch process that runs periodically: the frequency of refreshes and updates is usually dependent on the design of the crawler
- Retrieval is an online problem that demands sub-second response time.
- Individual users expect low query latencies, but query throughput is equally important since a retrieval engine must usually serve many users concurrently.

Web Crawling

- Gathering web content and building inverted indexes are for the most part offline problems.
- However, effective and efficient web crawling is far more far more complex. The following lists a number of issues that real-world crawlers must contend with:
- 1. A web crawler must practice good “etiquette" and not overload web servers. For example, it is common practice to wait a fixed amount of time before repeated requests to the same server. In order to respect these constraints while maintaining good throughput, a crawler typically keeps many execution threads running in parallel and maintains many TCP connections (perhaps hundreds) open at the same time.

Web Crawling

- 2. Since a crawler has finite bandwidth and resources, it must prioritize the order in which unvisited pages are downloaded. Such decisions must be made online and in an adversarial environment, in the sense that spammers actively create “link farms" and “spider traps" full of spam pages to trick a crawler into overrepresenting content from a particular site.
- 3. Most real-world web crawlers are distributed systems that run on clusters of machines, often geographically distributed. To avoid downloading a page multiple times and to ensure data consistency, the crawler as a whole needs mechanisms for coordination and load-balancing. It also needs to be robust with respect to machine failures, network outages, and errors of various types.

Web Crawling

- 4. Web content changes, but with different frequency depending on both the site and the nature of the content. A web crawler needs to learn these update patterns to ensure that content is reasonably current. Getting the right recrawl frequency is tricky: too frequent means wasted resources, but not frequent enough leads to stale content.
- 5. The web is full of duplicate content. Examples include multiple copies of a popular conference paper, mirrors of frequently-accessed sites such as Wikipedia, and newswire content that is often duplicated. The problem is compounded by the fact that most repetitious pages are not exact duplicates but near duplicates (that is, basically the same page but with different ads, navigation bars, etc.) It is desirable during the crawling process to identify near duplicates and select the best exemplar to index.

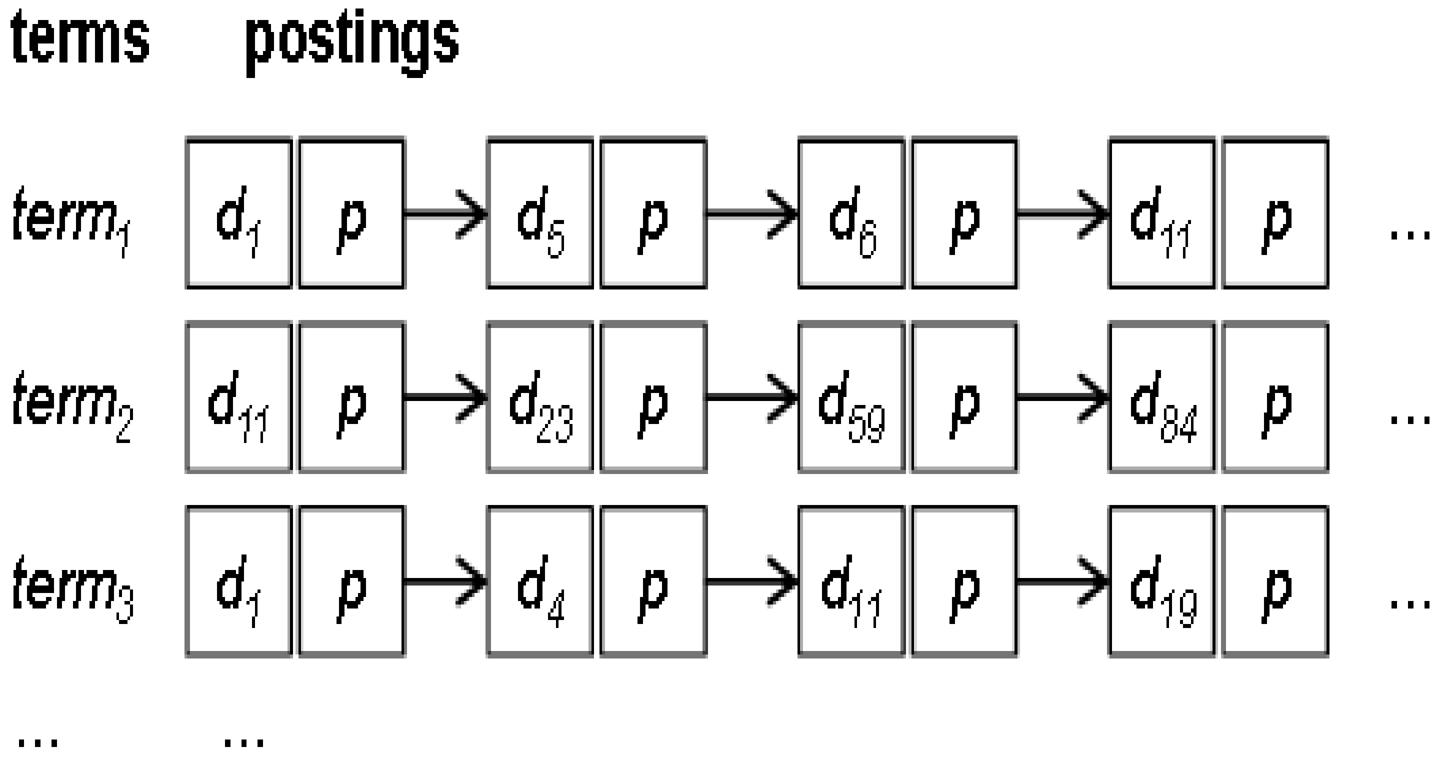
Web Crawling

- 6. The web is multilingual. There is no guarantee that pages in one language only link to pages in the same language. For example, a professor in Asia may maintain her website in the local language, but contain links to publications in English. Furthermore, many pages contain a mix of text in different languages. Since document processing techniques (e.g., tokenization, stemming) differ by language, it is important to identify the (dominant) language on a page.

Inverted Index

- An inverted index consists of postings lists, one associated with each term that appears in the collection.
- A postings list is comprised of individual postings, each of which consists of a document id and a payload -- information about occurrences of the term in the document.
- The simplest payload is. . . Nothing
- For simple Boolean retrieval, no additional information is needed in the posting other than the document id; the existence of the posting itself indicates that presence of the term in the document.

Inverted Index



Inverted Index

- The most common payload, however, is term frequency (tf), or the number of times the term occurs in the document.
- More complex payloads include positions of every occurrence of the term in the documents, properties of the term, or even the results of additional linguistic processing.
- Given a query, retrieval involves fetching postings lists associated with query terms and traversing the postings to compute the result set.
- In the simplest case, Boolean retrieval involves set operations (union for Boolean OR and intersection for Boolean AND) on postings lists, which can be accomplished very efficiently since the postings are sorted by document id.

Inverted Index

- In the general case, however, query - document scores must be computed.
- Partial document scores are stored in structures called accumulators.
- At the end the top k documents are then extracted to yield a ranked list of results for the user. Of course, there are many optimization strategies for query evaluation (both approximate and exact) that reduce the number of postings a retrieval engine must examine.
- The size of an inverted index varies, depending on the payload stored in each posting.

Inverted Index using MapReduce : Baseline Inverted Index

- Input to the mapper consists of document ids (keys) paired with the actual content (values).
- Individual documents are processed in parallel by the mappers.
- Each document is analyzed and broken down into its component terms.
- Once the document has been analyzed, term frequencies are computed by iterating over all the terms and keeping track of counts.
- Lines 4 and 5 in the pseudo-code reflect the process of computing term frequencies, but hides the details of document processing

Inverted Index using MapReduce : Baseline Inverted Index

- After this histogram has been built, the mapper then iterates over all terms.
- Each term, a pair consisting of the document id and the term frequency is created.
- Each pair, denoted by {n;H{t}} in the pseudo-code, represents an individual posting.
- The mapper then emits an intermediate key-value pair with the term as the key and the posting as the value, in line 7 of the mapper pseudo-code.

Inverted Index using MapReduce : Baseline Inverted Index

- In the shuffle and sort phase, the MapReduce runtime essentially performs a large, distributed group by of the postings by term.
- Without any additional effort by the programmer, the execution framework brings together all the postings that belong in the same postings list.
- This tremendously simplifies the task of the reducer, which simply needs to gather together all the postings and write them to disk.
- The reducer begins by initializing an empty list and then appends all postings associated with the same key (term) to the list.
- The postings are then sorted by document id, and the entire postings list is emitted as a value, with the term as the key.

Inverted Index using MapReduce : Baseline Inverted Index

```
1: class Mapper
2:     procedure Map(docid $n$, doc $d$)
3:         $H \leftarrow$ new AssociativeArray
4:         for all term $t \in$ doc $d$ do
5:             $H\{t\} \leftarrow H\{t\} + 1$
6:         for all term $t \in H$ do
7:             Emit(term $t$, posting $\langle n, H\{t\}\rangle$)
```

```
1: class Reducer
2:     procedure Reduce(term $t$, postings $[\langle n_1, f_1\rangle, \langle n_2, f_2\rangle \ldots]$)
3:         $P \leftarrow$ new List
4:         for all posting $\langle a, f\rangle \in$ postings $[\langle n_1, f_1\rangle, \langle n_2, f_2\rangle \ldots]$ do
5:             Append($P$, $\langle a, f\rangle$)
6:         Sort($P$)
7:         Emit(term $t$, postings $P$)
```

Inverted Index using MapReduce : RevisedBaseline Inverted Index

- There is a significant scalability bottleneck is available in Baseline Inverted Index algorithm.
- The algorithm assumes that there is sufficient memory to hold all postings associated with the same term.
- For efficient retrieval, postings need to be sorted by document id.
- The basic MapReduce execution framework makes no guarantees about the ordering of values associated with the same key, the reducer first buffers all postings and then performs an in-memory sort before writing the postings to disk.
- However, as collections become larger, postings lists grow longer, and at some point in time, reducers will run out of memory.

Inverted Index using MapReduce : RevisedBaseline Inverted Index

- Solution to this problem:
- Since the execution framework guarantees that keys arrive at each reducer in sorted order, one way to overcome the scalability bottleneck is to let the MapReduce runtime do the sorting for us.
- Instead of emitting key-value pairs of the following type: **(term t, posting {docid, f})**, we emit intermediate key-value pairs of the type instead: **(tuple {t,docid}, tf f)**.
- That is the key is a tuple containing the term and the document ID, whereas the value is the term frequency .
- With this modification, the postings are arrives in the correct order.
- This, combined with the fact that reducers can hold state across multiple keys, allows postings lists to be created with minimal memory usage.
- So, we must define a **custom partitioner** to ensure that all tuples with the same term are shuffled to the same reducer.

Inverted Index using MapReduce : RevisedBaseline Inverted Index

```
class Mapper
    method Map(docid $n$, doc $d$)
        $H \leftarrow$ new AssociativeArray
        for all term $t \in$ doc $d$ do
            $H\{t\} \leftarrow H\{t\} + 1$
        for all term $t \in H$ do
            Emit(tuple $\langle t, n \rangle$, tf $H\{t\}$)
```

```
class Reducer
    method Initialize
        $t_{prev} \leftarrow \emptyset$
        $P \leftarrow$ new PostingsList
    method Reduce(tuple $\langle t, n \rangle$, tf $[f]$)
        if $t \neq t_{prev} \wedge t_{prev} \neq \emptyset$ then
            Emit(term $t$, postings $P$)
            $P$.Reset()
        $P$.Add($\langle n, f \rangle$)
        $t_{prev} \leftarrow t$
    method Close
        Emit(term $t$, postings $P$)
```

Inverted Index using MapReduce : RevisedBaseline Inverted Index

- Solution to this problem:
- Since the execution framework guarantees that keys arrive at each reducer in sorted order, one way to overcome the scalability bottleneck is to let the MapReduce runtime do the sorting for us.
- Instead of emitting key-value pairs of the following type: **(term t, posting {docid, f})**, we emit intermediate key-value pairs of the type instead: **(tuple {t,docid}, tf f)**.
- That is the key is a tuple containing the term and the document ID, whereas the value is the term frequency .
- With this modification, the postings are arrives in the correct order.
- This, combined with the fact that reducers can hold state across multiple keys, allows postings lists to be created with minimal memory usage.
- So, we must define a **custom partitioner** to ensure that all tuples with the same term are shuffled to the same reducer.