Digital Libraries

Orientation

Week 1 Min-Yen KAN

What is a library?

1. A place set apart to contain books for reading, study, or reference.
   - (Not applied, e.g. to the shop or warehouse of a bookseller.)
2. A building … containing a collection of books for the use of the public or of some particular portion of it, or of the members of some society or the like;
3. a public institution or establishment, charged with the care of a collection of books, and the duty of rendering the books accessible to those who require to use them.

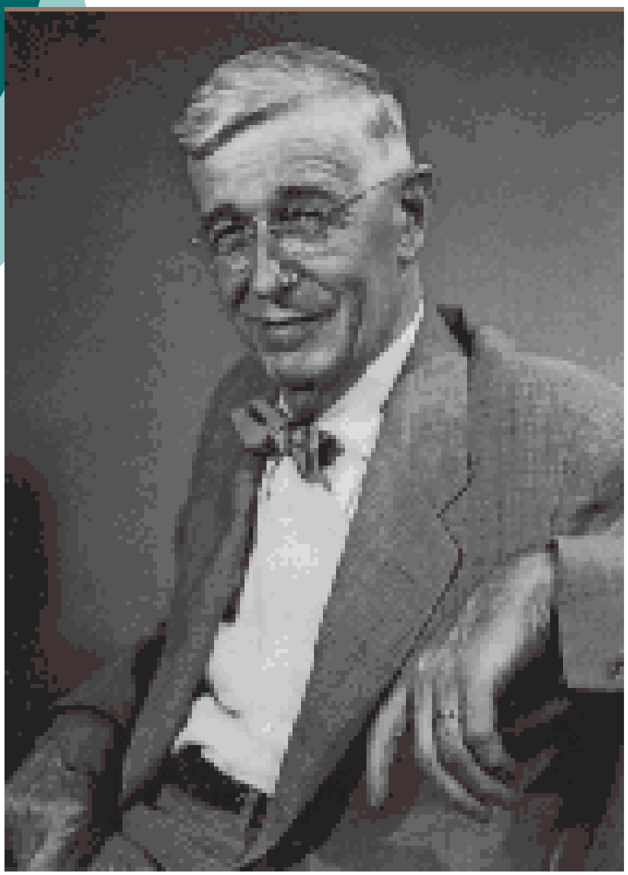
What is a library?

1. A private commercial establishment for the lending of books, the borrower paying either a fixed sum for each book lent or a periodical subscription.
2. a great mass of learning or knowledge;
3. the objects of a person's study, the sources on which he depends for instruction.
4. *Computers*. An organized collection of routines, esp. of tested routines suitable for a particular model of computer
5. *Biology*. a collection of sequences of DNA … that represent the genetic material of a particular organism or tissue

Introduction

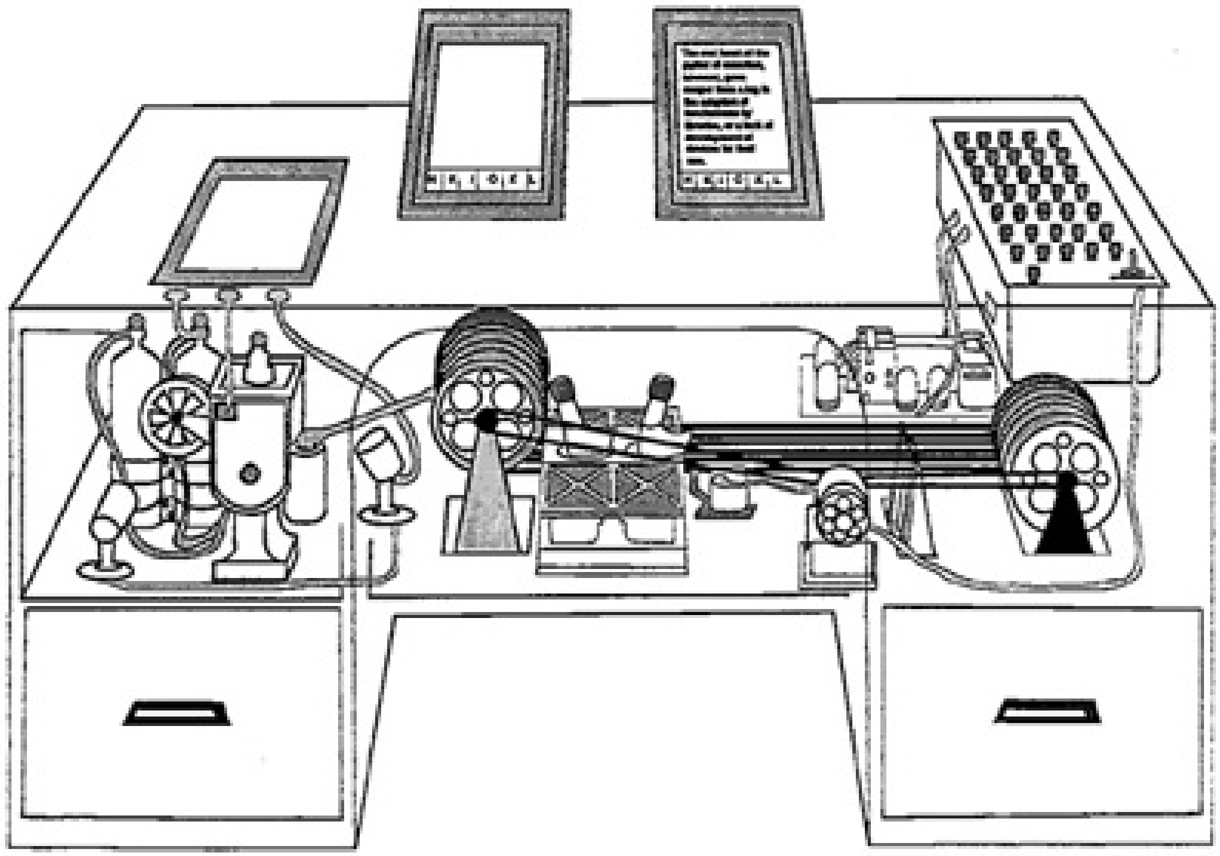
- Bush's "As we may think"
  - Writes this at the end of WW II
  - ______ was the first computer, born to compute ballistic tables fast
  - ________ just invented 5 years ago
  - ________ ("display technology") still a less than perfect process.
  - ________ ("storage technology") was a mature and stable technology.

Vannevar Bush (1890-1974)



- Director of the Office of Scientific Research and Development
  - lead 6000 scientists in R&D for WWII
- Predicted many technological advances
  - the "memex" is one whose spirit we are implementing
  - the purpose was to provide scientists the capability to exchange information; to have access to the totality of recorded information

Design for Memex (c. 1945)



Memex

- Integrated computer, keyboard, and desk
- "mechanized private file and library"
  - remove drudgery from information retrieval
  - suggested implementation was microfilm
  - various user operations are suggested
- ____________________ was the main purpose
  - "the process of tying two items together is the important thing"
  - prelude to hypertext...

Memex

- Information could come pre-associatively indexed, but the key point was ________________
  - WWW still does not provide that today
- Bush observes that tools change our way of doing, and expand the horizons before us
  - full impact of WWW and DLs still not known

What is a Digital Library (DL)?

- "a collection of information that is both digitized and organized" (Lesk)
  - there are numbers of alternate definitions, but this seems fair enough
  - no mention of _________, __________, ___________, etc.

- It is not just to reform the current library system, rather, we aim to
  - organize and access the "information overload"

Outline for today

- Introduction to libraries √
- Course administration
- Reading and writing research
- To think about

Course administration

- Teaching staff
- Web sites
- Objective
- Syllabus
- Assessment overview
- Survey paper and project

Any questions?

Teaching staff

- Lecturer:

  Min-Yen Kan ("Min")

  kanmy@comp.nus.edu.sg

  Office: S15 05-05

  6875-1885

  Hours: 4-6 pm Tuesdays

  Interests:
  rock climbing,
  ballroom dancing,
  and inline skating…
  and digital libraries!

Course web sites

http://ivle.nus.edu.sg/

- Discussion forum
  - Any questions related to the course should be raised on this forum
  - I expect you to talk amongst yourselves to answer questions, so will not answer questions here much.
  - Send me emails for urgent or personal matters
- Announcements!
- Workbin: Lecture notes (purposely incomplete!)

http://www.comp.nus.edu.sg/~cs5244

- Grading specification
- Other supplementary content

Objective

- Building, using, presenting and maintaining large volumes of information
- Contrast computational approaches with traditional library science methods

Hey min, go over the website!

- http://www.comp.nus.edu.sg/~cs5244

Discussions

Class participation is very important. There are no “dumb” questions. You will only be penalized for “no” questions / comments.

Possibilities:

- Name tags
- Cold calls
- Small group discussion and presentation

Midterm and Final

- 1 hour midterm (10%) and a 2 hour final (20%)
  - Both basically of the same format
  - Calculation questions – that have an exact answer
  - Essay questions – many to look at tradeoffs in the digital library realm
    - No necessarily right or wrong answers

Literature survey

- Each student will pick an area of study to survey at least **4** papers in detail.

- Must be **interesting** to you
- Journal or conference papers from an authority list
- Limit to 6 pages
- Individual work only
- Give your perspective on area's future
- Add value by comparing strengths and weaknesses of different approaches.

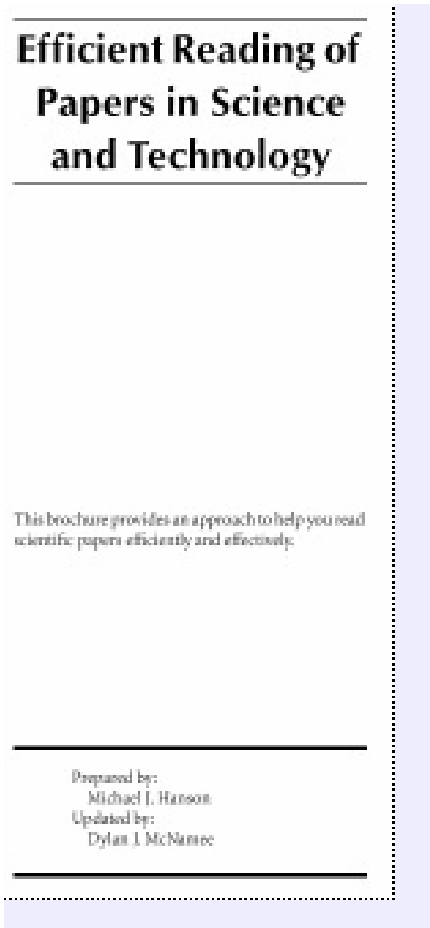
Final project

- Students will self-organize into groups for the final projects, shortly after the survey papers are due.

- Requires **original** work
- Cooperation and coordination
- Report as a conference submission
- Poster presentation to the public
- Sample topics on the web page

Outline for today

- Introduction to libraries √
- Course administration √
- Reading and writing research
- To think about



Efficient Reading of Papers in Science and Technology

This brochure provides an approach to help you read scientific papers efficiently and effectively.

Prepared by:
Michael J. Hanson
Updated by:
Dylan J. McNamee

Reading and writing research papers

References:

- http://www.cse.ogi.edu/~dylan/efficientReading.html
- ftp://fast.cs.utah.edu/pub/writing-papers.ps

This section partially from Surendar Chandra of University of Notre Dame.

Why do you read a paper?

- Understand and learn new contributions

- However…
  - Not all papers are “good”
  - Not all papers are “interesting”
  - Not all papers are “worthwhile” for you

- You have to learn to identify a good paper and spend your time wisely
  1. Breadth
  2. Depth
  3. React

Reading a research paper

- What is this paper about?
  1. Read the title and the abstract

     If you still don’t know what this paper is about, then this is a poorly-written paper.
  2. Read the conclusion

     Are you now sure you know what this paper is about? If not, throw it away.

3. Read the ______________
4. Read the _______________
5. Read _________________ and captions

How to read a paper

- See who wrote it, where it was published, when was it written (credibility)
- Skim references
  - Are authors are aware of relevant related work?
  - Do you know the work that they cite?
  - Do you know other work that they should have cited?

How to read a paper - depth

- Approach with scientific skepticism
- Read with context of other things that you've read in mind
  - It's only one part of the puzzle of a subject
- Examine the **assumptions**. Are they:
  - Reasonable?
  - What are the limitations of the work
    - There are always limitations! Did they disclose them?

How to read a paper - depth

- Examine the **methods**:
  - Did they measure what they claim?
  - Can they explain what they observed?
    - Want an analysis of **why** the system behaves a certain way, not raw data.
  - Did they have adequate controls?
  - Were tests carried out in a standard way? Were the performance metrics standard?
    - If not, do they explain their metrics clearly?

How to read a paper - depth

- Examine the **statistics**:
  "Lies, d*mned lies and statistics"
  - Appropriate statistical tests applied properly?
  - Did they do proper error analysis?
  - Are the results statistically significant?

How to read a paper - depth

- Examine the conclusions:
  - Do the conclusions follow logically from the experiments?
  - What other explanations are there for the observed effects ?
  - What other conclusions or correlations are in the data that were not pointed out?

How to read a paper - react

- Take notes
- Highlight major points
- React to the points in the paper
  - Place this work with your own experience
  - If you doubt a statement, note your objection

- **Summarize** what you read
  - Good practice: maintain your own bibliography of all papers that you ever read
  - _____________ !

How to write a research paper

- Write it such that anyone who reads it using the method we just discussed understands the idea

- Clearly explain what problem you are solving, why it is interesting and how your solution solves this interesting problem

- Be crisp. Explain what your contributions are, what your ideas are and what are others' ideas

Any questions?

Introduction to libraries √

Course administration √

Reading and writing research √

To think about for discussion

- What are the functions of a traditional library?
- Are these same functions in the **digital** library?
- How is the digital library different from:
  - ____________?
  - ____________?

Coffee Break

See ya!

Digital Libraries

Week 1 Min-Yen KAN

Implementation of
(Textual) Information Retrieval

What is information retrieval?

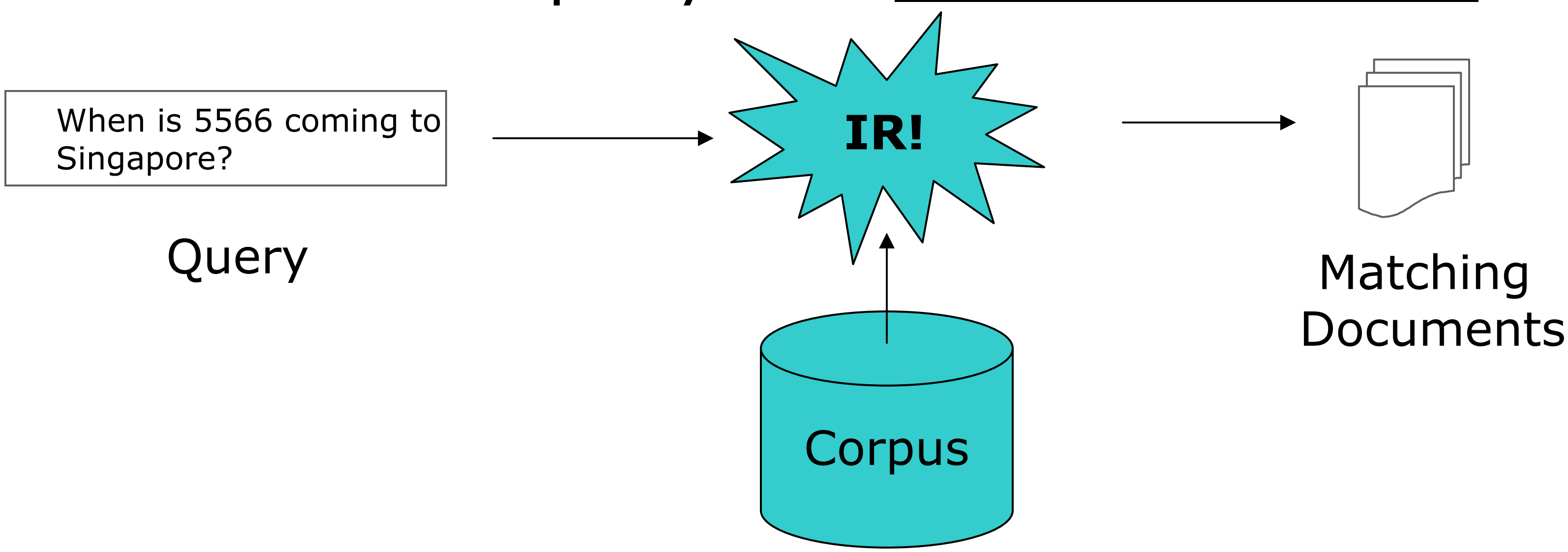
Midterm questions for Digital Libraries

Search

Search

What is information retrieval?

- Part of the information seeking process
- Matches a query with most relevant documents
- View a query as a ________________-



Searching in books

- ______
- ______
- ______

- Procedure:
  - Look up topic
  - Find the page
  - Skim page to find topic

…
Index, 11, 103-151, 443
  Audio, 476
  Comparison of methods 143-145
  Granularity, 105, 112
  N-gram, 170-172
  Of integer sequences, 11
  Of musical themes, 11
  Of this book, 103, 507*ff*
  Within inverted file entry, *see* skipping
Index compression, 114-129, 198-201, 235-237
  Batched, 125,128
  Bernoulli, 119-122, 128, 150, 247, 421
  Context-sensitive, 125-126
  Global, 115-121
  Hyperbolic model, 123-124, 150
  In MG, 421-423
  Interpolative coding, 126-128
  Local, 115, 121-122, 247
  Nonparameterized, 115-119
  Observed frequency, 121, 124-125, 128, 247
  Parameterized, 115
Performance of, 128-129. 421
Skewed Bernoulli, 122-123, 138, 150
Within-document frequencies, 198-201
Index Construction, 223-261 (*see also* inversion)
  bitmaps, 255-256
…

Partial index of *Managing Gigabytes*

Information retrieval

- Algorithm
  - (Permute query to fit index)
  - Search index
  - Go to resource
  - (Permute query to fit item)
  - (Search for item)

What to index?

- Books indices have key words and phrases
- Search engines index ______________

Why the disparity?

What do people really search for?

What is a **word**?

- Maximal sequence of alphanumeric characters
- Limited to at most 256 characters and at most 4 numeric characters.

- MG indexing system

Trading precision for size

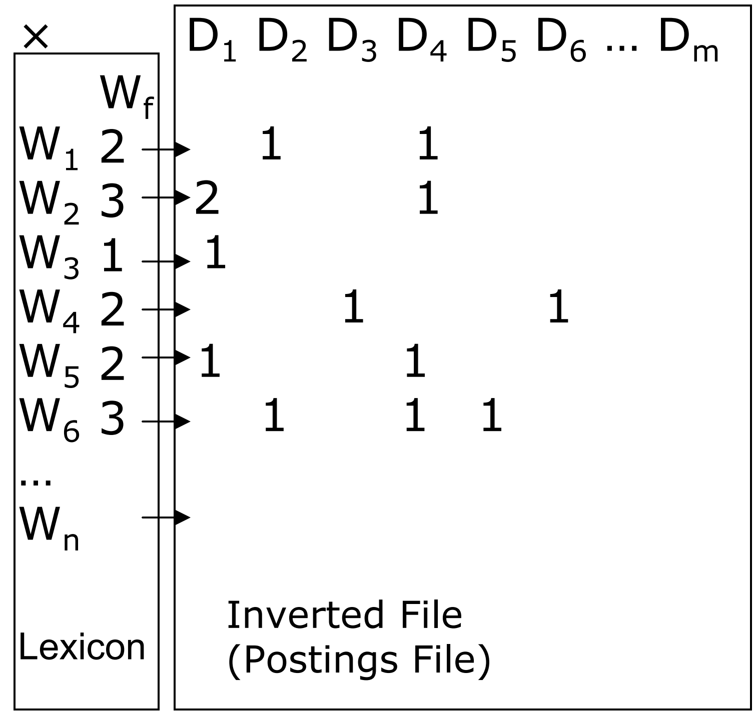
Can save up to **32%** without too much loss:

- Stemming
  - Usually just word inflection
  - Information → Inform = Informal, Informed
- Case folding
  - **N.B.**: keep odd variants (e.g., NeXT, LaTeX)
- Stop words
  - Don't index common words, people won't search on them anyways

Pop Quiz: Which of these techniques are more effective?

Indexing output

- Output = $L_w, D_D, I_{W \times D}$
- Inverted File (Index)
  - Postings (*e.g.*, $w_t \rightarrow (d_1, f_{wt,d1}), (d_2, f_{wt,d}), \ldots, (d_n, f_{wt,dn})$
  - Variable length records
- Lexicon:
  - String W_t
  - Document frequency f_t
  - Address within inverted file I_t
  - Sorted, fixed length records



To think about: What type of entries are missing from the search engine index that are present in the book index?

Trading precision for size, redux

Pop Quiz: Which of these techniques are more effective?

Typical:
Lexicon = 30 MB   Inverted File: 400 MB

- Stemming
  - Affects Lexicon
- Case folding
  - Affects Lexicon
- Stop words
  - Affects Inverted File

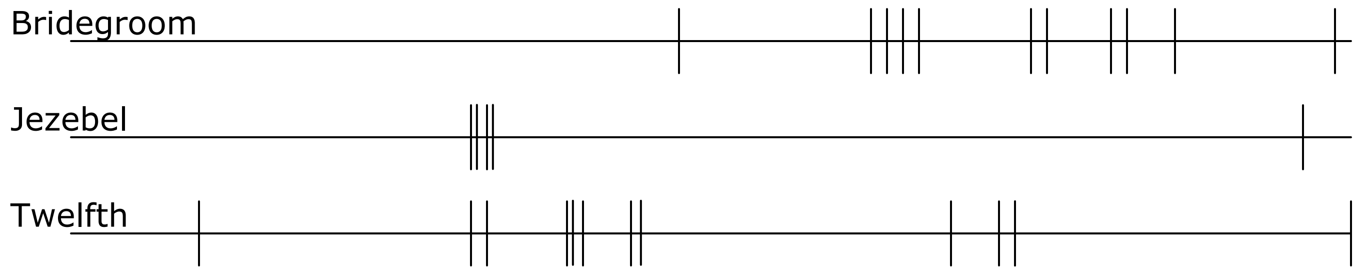
Is fine-grained indexing worthwhile?

- **Problem**: still have to scan document to find the term.

Image	(D1, 2), (D4, 1)	➡	Image	(D1, 2; 10, 205), (D4, 1, 3993)
Implicit	(D2, 1), (D3, 1) …		Implicit	(D2, 1; 242), (D3, 1; 233) …
Index	(D5, 3), (D2, 1) …		Index	(D5, 3; 20, 42, 3920), (D2, 1 …
Inverse	(D2, 2)		Inverse	(D2, 2; 599, 847)
Internet	(D1, 2), (D3, 2) …		Internet	(D1, 2; 12, 43), (D3, 2; 302, …

- Cons:
  - Need access methods to take advantage
  - Extra storage space overhead (variable sized)
- Alternative methods:
  - Hierarchical encoding (doc #, para #, sent #, word #) to shrink offset size
  - Split long documents into *n* shorter ones.

Inverted file compression



Occurrences in the *Bible*

- Clue: Encode *gap length* instead of offset
- Use small number of bits to encode more common gap lengths
  - (e.g., Huffman encoding)
- Better: Use a distribution of expected gap length (e.g., Bernoulli process)
  - If p = prob that any word x appears in doc y, then
  - Then $p_{\text{gap size } z} = (1-p)^z p$. This constructs a geometric distribution.

- Works for intra and inter-document index compression
  - Why does it hold for documents as well as words?

Building the index – Memory based inversion

Initialize empty dictionary S
// Phase I – collection of term appearances in memory
For each document D_d in collection, $1 \leq d \leq N$
    Read D_d, parsing it into index terms
    For each index term t in D_d
        Calculate $f_{d,t}$
        Search in S for t, if not present, insert it
        Append node $(d,f_{d,t})$ to list for term t

// Phase II – dump inverted file
For each term $1 \leq t \leq n$
    Start a new inverted file entry
        Append each appropriate $(d,f_{d,t})$ in list to entry
    Append to inverted file

- Takes lots of main memory, ugh!
- Can we reduce the memory requirement?

Sort-based inversion

- **Idea**: try to make random access of disk (memory) sequential

```
// Phase I – collection of term appearances on disk
For each document D_d in collection, 1 ≤ d ≤ N
      Read D_d, parsing it into index terms
      For each index term t in D_d
            Calculate fd,t
            Dump to file a tuple (t,d,f_d,t)
// Phase II – sort tuples
Sort all the tuples (t,d,f) using External Mergesort

// Phase III – write output file
Read the tuples in sorted order and create inverted file
```

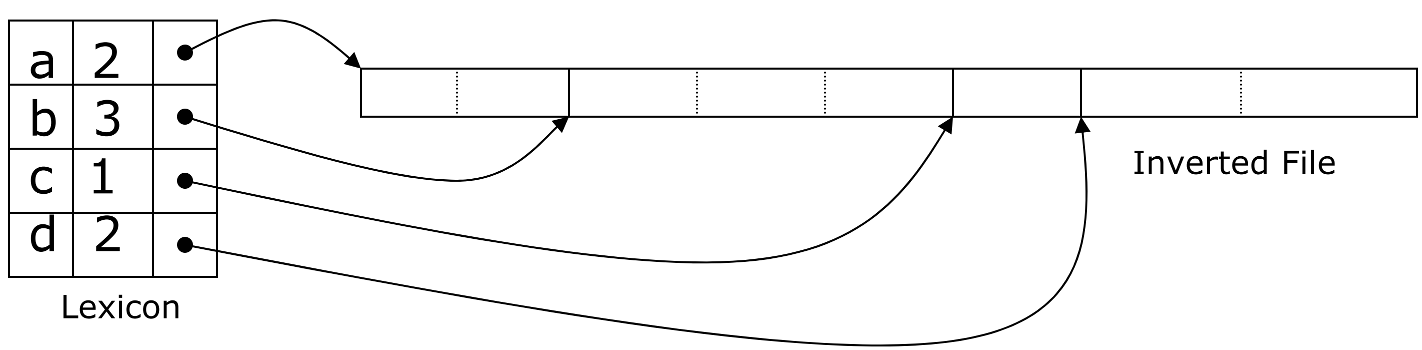
Sort based inversion: example

Initial dump from corpus	Sorted Runs	Merged Runs (fully sorted)
<a,1,2>	<a,1,1>	<a,1,1>
<b,1,2>	<a,2,2>	<a,2,2>
<c,1,1>	<b,1,2>	<b,1,2>
<a,2,2>	<c,1,1>	<b,2,1>
<d,2,1>	<b,2,1>	<b,3,1>
<b,2,1>	<b,3,1>	<c,1,1>
<b,3,1>	<d,2,1>	<d,2,1>
<d,3,1>	<d,3,1>	<d,3,1>

- What's the performance of this algorithm?
- Saves memory but very disk intensive!

Using a first pass for the lexicon

- Gets us $f_{d,t}$ and N
  - **Savings**: For any t, we know $f_{d,t}$, so can use an array vs. LL (shrinks record by 40%!)



Lexicon-based inversion

- Partition inversion as |I|/|M| = k smaller problems
  - build 1/k of inverted index on each pass
  - (e.g., a-b, b-c, …, y-z)
  - Tuned to fit amount of main memory in machine
  - Just remember *boundary words*
- Can pair with disk strategy
  - Create k temporary files and write tuples $(t,d,f_{d,t})$ for each partition on first pass
  - Each second pass builds index from temporary file

Inversion – Summary of Techniques

- How do these techniques stack up?
- Assume a 5 GB corpus and 40 MB main memory machine

Technique	Memory (MB)	Disk (GB)	Time (Hours)
*Linked lists (memory)	4000	0	6
Linked lists (disk)	30	4	1100
Sort-based	40	8	20
Lexicon-based	40	0	79
Lexicon w/ disk	40	4	12

Source – Managing Gigabytes

Query Matching

Now that we have an index, how do we answer queries?

Query Matching

Assuming a simple word matching engine:

```
For each query term t
      Stem t
      Search lexicon
      Record f_t and its inverted entry address, I_t
Select a query term t
Set list of candidates, C = I_t
For each remaining term t
      Read its I_t
      For each d in C, if d not in I_t set C = C – {d}
```

Conjunctive (AND) processing

- X and Y and Z – high ________
- X or Y or Z – high ________
- Which algorithm is the above?

Boolean Model

- Query processing strategy:
  - Join less frequent terms first
  - Even in ORs, as merging takes longer than lookup
- Problems with Boolean model:
  - Retrieves too many or too few documents
  - Longer documents are tend to match more often because they have a larger vocabulary
  - Need ranked retrieval to help out

Deciding ranking

- Boolean assigns same importance to all terms in a query

5566 concert dates in Singapore — Search

  - "5566" has same weight as "date"

- One way:
  - Assign weights to the words, make more important words worth more
  - Process results in *q* and *d* vectors: (word, weight), (word, weight) … (word, weight)

Term Frequency

Xxxxxxxxxxxxxxx IBM xxxxxxxxxxxx xxxxxxxxx xxxxxxxxxxxx IBM xxxxxxx xxxxxxxxxxx xxxxxxxxx Apple. Xxxxxxxxxxx xxxxxxxxxxx IBM xxxxxxxx. Xxxxxxxxxxx xxxxxxxxx Compaq. Xxxxxxxxxx xxxxxxx IBM.

(Relative) term frequency can indicate importance.

- $R_{d,f} = f_{d,t}$
- $R_{d,t} = 1 + \ln f_{d,t}$
- $R_{d,t} = (K + (1-K) \frac{f_{d,t}}{\max_i(f_{d,i})})$

Inverse Document Frequency

Consider a future device for individual use, which is a sort of mechanized private file and library. It needs a name, and, to coin one at random, "memex" will do.

Inverse Document Frequency

Consider a future **device** for **individual** use, which is a sort of **mechanized private** file and **library**. It needs a name, and, to coin one at **random**, "**memex**" will do.

- Words with higher f_t are less discriminative.
- Use inverse to measure importance:
  - $w_t = 1/f_t$
  - $w_t = \ln(1+ N/f_t)$ ← this one is most common
  - $w_t = \ln(1 + f^m/f_t)$, where f^m is the max observed frequency

Question: What's the ln () here for?

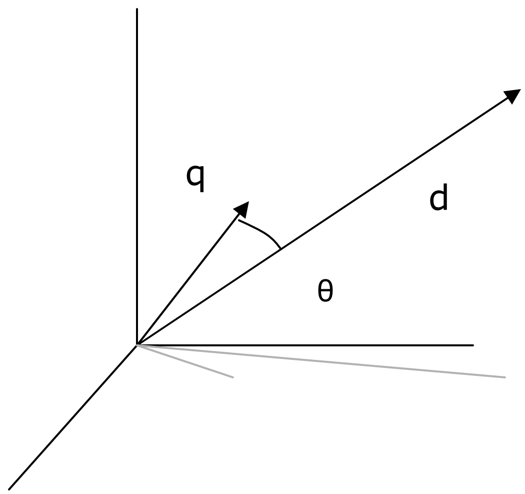
This is TF*IDF

- Many variants, but all capture:
  - Term frequency:
    $R_{d,t}$ as being ______________________
  - Inverse Document Frequency:
    W_t as being _______________________
- Standard formulation is:
  $w_{d,t} = r_{d,t} \times w_t$
  $= (1 + \ln(f_{d,t})) \times \ln(1 + N/f_t)$
- Problem:
  - $r_{d,t}$ grows as document grows, need to normalize; otherwise biased towards ________________

Calculating Similarity

- Euclidean Distance - bad
  - $M(Q,D_d) = \text{sqrt}\ (\Sigma\ |w_{q,t} - w_{d,t}|^2)$
  - Dissimilarity Measure; use reciprocal
  - Has problem with long documents, **why**?
- Actually don't care about vector length, just their direction
  - Want to measure difference in direction

Cosine Similarity



- If X and Y are two *n*-dimensional vectors:

  $X \cdot Y = |X|\ |Y| \cos \theta$

  $\cos \theta = X \cdot Y / |X|\ |Y|$

  = 1 when identical

  = 0 when orthogonal

$$\text{Cos}(Q, Dd) = Q \cdot D_d / |Q|\ |D_d|$$
$$= (1/W_q W_d) \Sigma\, w_{q,t} \cdot w_{d,t}$$
$$= (1/W_d) \Sigma\, w_{q,t} \cdot w_{d,t}$$

Calculating the ranked list

$$\frac{1}{W_d W_q} \sum_{t \in Q \cap D_d} (1 + \ln f_{d,t}) \bullet \ln(1 + \frac{N}{f_t})$$

- To get the ranked list, we use doc. accumulators:

For each query term t, in order of increasing f_t,
  Read its inverted file entry I_t
  Update acc. for each doc in I_t: A_d+= ln (1 + $f_{d,t}$) ×w_t
For each A_d in A
  A_d /= W_d **// that's basically cos θ, don't use w_q**
Report top r of A

Accumulator Storage

- Holding all possible accumulators is expensive
  - Could need one for each document if query is broad
- In practice, use fixed |A| wrt main memory. What to do when all used?
  - Quit: ____________________
  - Continue _________________________

Selecting r entries from accumulators

- Want to return documents with largest cos values.
- How? Use a min-heap

```
Load r A values into the heap H
Process remaining A-r values
  If A_d > min{H} then
    Delete min{H}, add A_d, and sift
// H now contains the top r exact cosine values
```

To think about

- How do you deal with a dynamic collection?
- How do you support phrasal searching?
- What about wildcard searching?
  - What types of wildcard searching are common?