CS657: Information Retrieval
Boolean Retrieval Model

Arnab Bhattacharya
arnabb@cse.iitk.ac.in

Computer Science and Engineering,
Indian Institute of Technology, Kanpur
http://web.cse.iitk.ac.in/~cs657/

2nd semester, 2017-18

Tue, Wed 1200-1315 at KD101

Document Retrieval Task

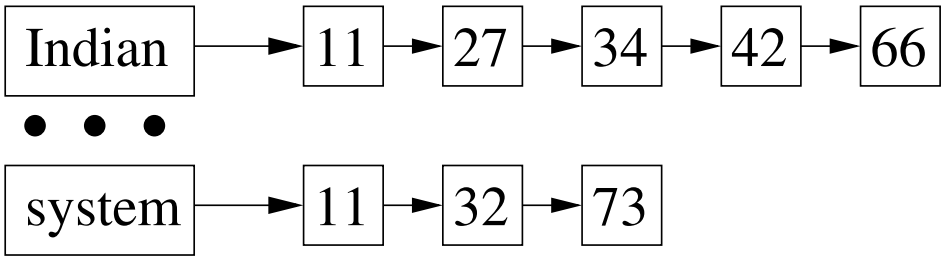
- Assume a set $\mathcal{D} = \{D_1, \ldots, D_m\}$ of *documents*
- Called a corpus or collection of documents
- Each D_i is composed of a number of *terms* $D_i = \{t_{i_j}\}$
- Bag-of-words model
  - No order
  - Not a multi-set
- The total collection of terms is called the dictionary or lexicon or vocabulary
- (Ad hoc) Boolean information retrieval task
  - Given a set of terms from the vocabulary, find the documents that contain them
  - Sense is generally AND
  - May sometimes involve OR or NOT
  - May even involve NEAR (requires order)
- Query of terms approximates *information need*

Methods for Answering

- Unix utility `grep` solves it nicely
- Problems
  - Very large collections
  - Distributed setup
  - Order of words (NEAR queries)
  - Ranking of results
- Boolean bit matrix for each document and each term
- Called incidence matrix
- Boolean algebra on the bit vectors of the queried terms
- Problems
  - Extremely sparse
  - Wastes space
  - All the other problems of `grep`
- *Not* scalable

Inverted Index

- "Invert" the sense: let terms be composed of documents



- Each term contains a list of documents (called postings) that it appears in
  - May also include the number of times it appears in a document, called term frequency
- This list of documents is called a postings list
  - Its size, called document frequency, is the number of documents a term appears
- The set of lists for all the terms is called postings
  - Its size is size of the vocabulary
- Postings list is maintained as a linked list

Querying using Inverted Index

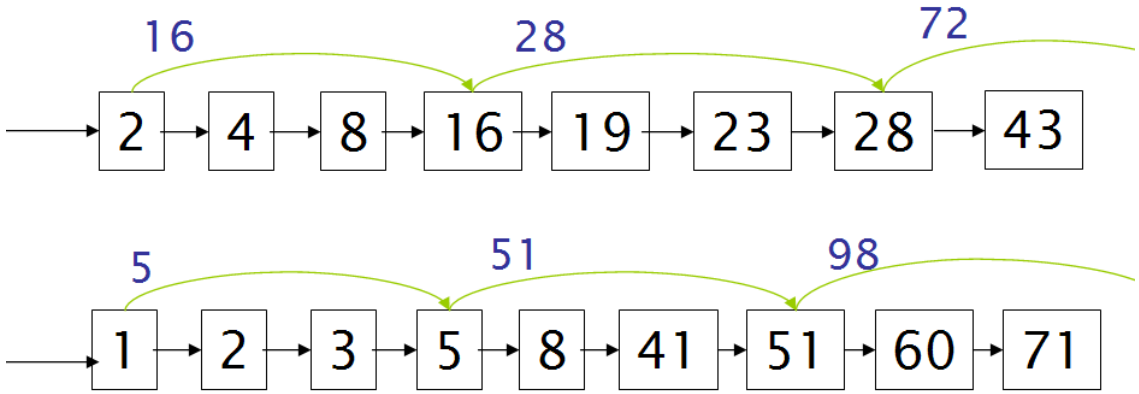
- Find documents where terms "Indian" and "system" occur
- Get corresponding postings lists
- *Intersect* them
- For two lists of size x and y, time required is $O(x.y)$
- If lists are sorted by document ids
  - *Merging* as in mergesort
  - Two pointers to walk along the lists
  - Time complexity is $O(x + y)$
- For multiple query terms: "ancient", "Indian" and "system"
  - Start with terms having *smallest* postings lists
  - Progressively merge
  - Size of final answer set is *never* more than any intermediate step
- OR is more time consuming
- Query with only NOT is impractical for large collections
  - All right with AND: "system" AND NOT "Indian"

Implementation

- Terms in memory
- Postings list on disk
- Terms are hashed
- Space requirements
  - Assume 2×10^6 terms
  - Each term is of length 8 characters
  - Each character requires 2 bytes
  - In addition, there is a disk pointer of size 8 bytes
  - Therefore, total of 24 bytes per term
  - Total term size is 48×10^6 bytes, i.e., in the order of 100 MB
  - Postings list can be 100 times more
- To save disk access
  - Postings lists can be compressed
  - Documents can be clustered
  - Partial results can be retrieved

Skip Lists

- Skip lists are used to traverse linked lists faster



- "Skips" are provided as jumps
- Useful when taking intersection of postings lists
- Once 41 is reached in the second list, 16 can jump to 28 in the first
- How to determine skip positions?
- Look at closely occurring values; insert a skip to the end
- $\sqrt{l}$ equally spaced skips for a l-length list
- Uses more space

Tokenization

- Terms are useful semantic units that need to be indexed for searching
- Tokenization is the process of breaking the text into *terms*

Where the Mind is Without Fear

```
Where the Mind is Without Fear
------------------------------
-- Rabindranath Thakur
Where the mind is without fear and the head is held high;
Where knowledge is free
Where the world has not been broken up into fragments
By narrow domestic walls
Where words come out from the depth of truth;
Where tireless striving stretches its arms towards perfection;
Where the clear stream of reason has not lost its way
Into the dreary desert sand of dead habit;
Where the mind is led forward by thee
Into ever-widening thought and action;
Into that heaven of freedom, My Father, let my country awake.
```

English

- Identifying words is almost trivial
- Idioms
  - "hot potato", "couch potato"
- Word markers or punctuations
  - "can't", "O'Neille"
- Hyphenation
  - "co-education", "e-mail"
- White space
  - "New Delhi", "de Villiers"
- Combinations
  - "isn't New Delhi-Uttar Pradesh a good example?"

Indian Languages

मूर्तमहेश्वरमुज्ज्वलभास्करमिष्टममरनरवन्दम्

मूर्त + महेश्वरम् + उज्ज्वल + भास्करम् + इष्टम् + अमर + नर + वन्दम्

- Sandhi: two words are joined together syntactically
- Samaas: two words are joined together semantically
- Upasarga: prefixes that alter the meaning

Indian Language Scripts

- Vowel marks are distinctive
- Different from stand-alone vowels
- Can change the consonant in different ways
  - क् +उ =कु but र् +उ =रु
  - क् +ऊ =कू but र् +ऊ = रू
- Vowel sign can appear before, after, over and/or under the consonant
  - ক্ + আ = কা
  - ক্ + এ = কে
  - ক্ + ও = কো
- Conjunct characters may look completely different:
  - क् +ष =क्ष
- Not a problem when UTF-8 is used

European Languages using Roman Script

- Use of diacritic marks: ö, á, à, etc.
- German is very prone to joining of words (*samaas*)
  - "computerlinguistik" (computational linguistics)
  - "gepaeckaufbewahrungstelle" (luggage supervision place)
- French merges the articles very often
  - "l'égalité (the equality)
- Even English
  - "automobile", "email"

Arabic

ك ِ ت ا ب ٌ ⇐ كِتابٌ

un b ā t i k

/kitābun/ *'a book'*

استقلت الجزائر في سنة 1962 بعد 132 عاما من الاحتلال الفرنسي.

← START ← → ← →

'Algeria achieved its independence in 1962 after 132 years of French occupation.'

- Bi-directionality should not be a problem with UTF-8 encoding

Chinese

和尚

- May mean “monk” (pronounced as “heshang”) or “and still”

莎拉波娃现在居住在美国东南部的佛罗里达。今年４月９日，莎拉波娃在美国第一大城市纽约度过了１８岁生日。生日派对上，莎拉波娃露出了甜美的微笑。

- No word segmentation at all
- Modern writings contain Indian (Arabic) numerals and Latin punctuation marks
- Same with Japanese, Korean and Thai

Japanese

ノーベル平和賞を受賞したワンガリ・マータイさんが名誉会長を務めるＭＯＴＴＡＩＮＡＩキャンペーンの一環として、毎日新聞社とマガジンハウスは「私の、もったいない」を募集します。皆様が日ごろ「もったいない」と感じて実践していることや、それにまつわるエピソードを８００字以内の文章にまとめ、簡単な写真、イラスト、図などを添えて１０月２０日までにお送りください。大賞受賞者には、５０万円相当の旅行券とエコ製品２点の副賞が贈られます。

- Uses 4 different alphabets
  - Traditional Chinese characters for normal words
  - *Hiragana* for inflectional endings and functional words
  - *Katakana* for transcription of foreign words
  - Indian numerals for numbers
- More importantly, query may be entirely in Hiragana, since that is easier to type

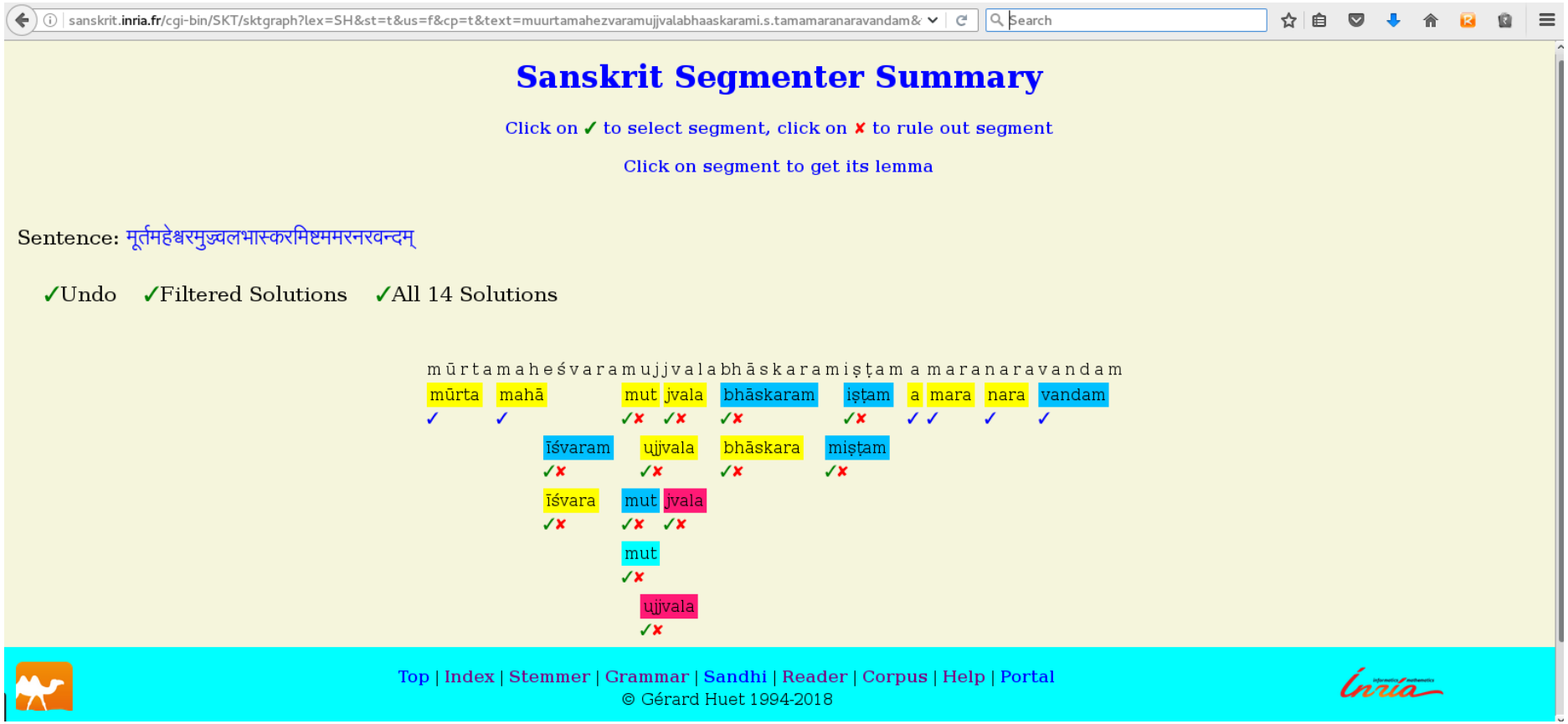
Language Identification

- *Language specific* tokenization rules
- *Language identification*
- *Script identification*
- Especially since modern writings involve multiple scripts and languages
- Unicode solves it to a large extent, but *not* completely
  - Bangla versus Ahomiya versus Manipuri, Marathi versus Hindi
- May require dictionaries

Compound Splitter

- Indian languages, German, etc. use a large number of compound words
- It is less useful to index them as-is
- *Compound splitter* module
- Task is to break a compound word into component parts such that each part makes sense
  - Is part of a dictionary
  - Dictionary cannot contain all inflectional forms
- Generally uses a lot of language specific grammar features
- Limit is reached for CJK, Thai, etc.
- Alternatively, forget about word boundaries altogether
  - "altogether = all + together"
- Use character k-grams
  - "whitespace" or "white space"
- Index every k consecutive characters as a *token*

Samskrit Word Splitter



Stop Words

- Stop words (or stopwords) are those having little value for the task at hand
- Generally, those having little *semantic* content
  - Simple keyword searching does not use semantics
- Words are sorted by frequency in the *corpus*
- Diversity of corpus is extremely important
  - Using only newspapers, high frequency words in Hindi are भारत, सरकार, दिल्ली, पुलिस, etc.
- Top ones are *hand-filtered* to produce stopword list
  - English: a, an, and, ...
- Help of *linguists* required
- Stopword removal greatly affects key phrase searching
  - "Captain of India", "to be or not to be"
- Indian languages do not generally have stopwords
  - "of", "in", etc. are vibhaktis and become part of the word: ভারতে, ভারতের, भारतात, भारताचा
- Web search engines do not bother to remove stopwords
  - Term weighting, compression, etc. help anyway

Token Normalization

- Token normalization (or term normalization) aims to match despite superficial differences
- Construct equivalence classes
- English
  - Hyphens: "e-mail" vs. "email"
  - Diacritics: "naive" vs. "naïve"
- Accents and diacritics are very important in some languages
  - Spanish: "peña" (cliff) vs. "pena" (sorrow)
  - Romanized Samskrit: स्वजनाः (svajanāḥ) vs. श्वजना: (śvajanāḥ)
  - German: *"Gauss"* is same as *"Gauβ"* (Eszett character)
- Abbreviations and short forms
  - "U.P." vs. "UP" (not "up"), "versus" vs. "vs."
- Spelling variants
  - "color" vs. "colour", "Chebyshev" vs. "Tchebycheff"
- Commas in numbers
  - European (1.23.45,56 euros) vs. American (1,23,45.56 dollars)
  - Indian (12,34,567) vs. American (1,234,567)
- Date formats
  - 23/01/1897 vs. 1-23-97 vs. 23rd January, 1897 vs. ...

Cases

- Indian languages admiringly do not use cases
- English, unfortunately, depends a lot on that
- Case folding reduces everything to lower case
  - "UP" $\rightarrow$ "up", "PIN" $\rightarrow$ "pin"
  - "Reliance" is definitely not "reliance"
- True-casing
  - Only letters in beginning of sentence are lower-cased
  - Words in the middle are left as is
- Again, intent depends on query
- May require *asymmetric query expansion*
  - window $\rightarrow$ window, windows
  - windows $\rightarrow$ windows, Windows
  - Windows $\rightarrow$ Windows

Lemmatization

- To reduce inflectional forms
- Stemming is chopping inflections
  - "computers" $\rightarrow$ "computer"
- Lemmatization is reducing to root word (lemma) by using *morphological analysis*
  - "are", "is", "am" $\rightarrow$ "be"
- Stemming and lemmatization can differ greatly
  - "saw" is stemmed to "s"
  - "saw" is lemmatized to "see" for verb and "saw" for noun
- Phonetics help
  - *Soundex* algorithm

Stemmers

- Porter stemmer
- Lovins stemmer
- Paice/Husk stemmer
- Porter's algorithm
- Uses language-specific hand-coded rules

Rule			Example		
SSES	$\rightarrow$	SS	caresses	$\rightarrow$	caress
IES	$\rightarrow$	I	ponies	$\rightarrow$	poni
SS	$\rightarrow$	SS	caress	$\rightarrow$	caress
S	$\rightarrow$		cats	$\rightarrow$	cat

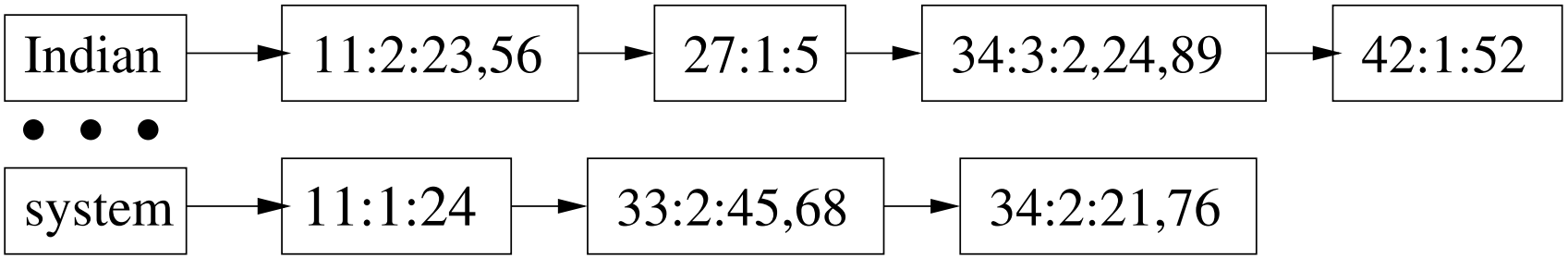
- 5 phases of reduction, in sequence
- Select rule that applies to longest suffix
- Rules may depend on chopped length
  - "replacement" $\rightarrow$ "replac", but "cement" $\not\rightarrow$ "c"
- Indian languages?
  - Samskrit: 1,70,000 words produce 11 million morphological forms (1:65)

Corpus-Based Stemmers

- Does not require language rules (mostly)
- Focus on suffixing languages
- Corpus should be large and diverse
- YASS: String-based
  - Tries to find largest common prefix
  - Finds edit distance between pairs of words
  - Penalizes early mis-matches more heavily
  - Identify longest common prefix as the lemmatized root
- GRAS: Graph-based
  - Identify common prefixes for a pair of words
  - Check if the suffix pair removed is valid
  - Suffix pair must be found in other word pairs
  - Model words by nodes connected by edges to other words sharing common prefix
  - Identify most common prefix as the lemmatized root

Position Queries

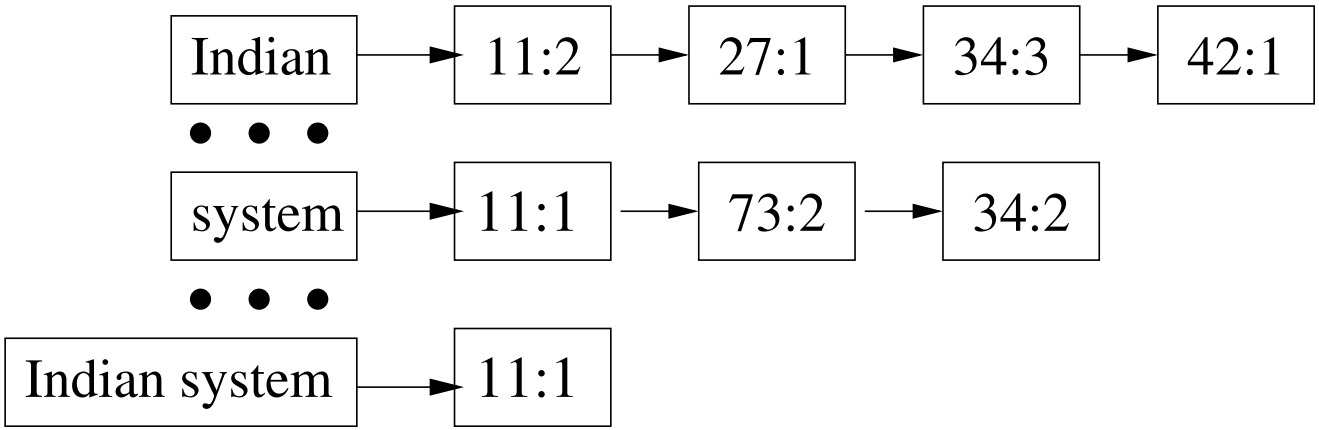
- NEAR queries: Find documents where "Indian" and "system" occur within 2 words of each other
- Positional postings



- PHRASE queries: "information retrieval"
- Idioms can be queried
- Positional postings require much more space

Multi-Word Indexes

- Bi-word index
- Each posting consists of only the frequency and not positions
- Generally used in addition to single word index



- Very useful for queries on names
  - "Subhas Chandra Bose"
- High storage
- Why not k-word index?
- Not really required
- Combination of positional and bi-word index

Spelling Errors and Variations

- How to search for both "colour" and "color"? "colo*r"
- Wildcard query
- Use special indexes to retrieve a larger set of candidates
- Check the candidates for answers
- Generic filter-and-prune paradigm

Permuterm Index

- Augment alphabet with a special end marker $
- Word "colour" becomes "colour$"
- Apply all rotations: "olour$c", "lour$co", "our$col", "ur$colo", "r$colou", and link them to "colour$"
- Suppose query is "colo*r"
- Augment with end marker to produce "colo*r$"
- Rotate to put '*' at end: "r$colo*"
- Search for all vocabulary terms linked to and starting with "r$colo"
- "r$colo" links to both "color" and "colour"
- What if query is "al*ha*et"?
- Search for both "al*" and "*et"
- Take intersection and check if those contain "ha"

K-Gram Index

- Suppose $k = 3$
- Word "colour" is broken into k-grams: "col", "olo", "lou", and "our"
- Link each k-gram to list of words where it appears
- "our" links to "colour", "flour", "hour", "our", etc.
- Suppose query is "c*our"
- Broken into 2 queries "c*" and "*our"
- Needs filtering later

Tries

- Comes from the word retrieval
- Mostly used for strings
- Structure of a *basic* trie
  - Root represents null string
  - Each edge defines the next character
  - Each node stores a string or a prefix of a string
  - Strings with same prefix share the path
- Trie for strings "steam", "string", "strong", "tea", and "team"