2017 Semester 2 Review

COMP3009J: Information Retrieval

Dr. David Lillis (david.lillis@ucd.ie)

UCD School of Computer Science
Beijing Dublin International College

Module Review

What is Information Retrieval?

- Find **information items** that help to satisfy the information need of a user.
- Query is a representation of an information need that is given to an IR system.
  - Types of queries, problems with querying.
  - **Example:** Boolean queries & incidence vectors.
- Role of the IR system: informs the user of the existence and whereabouts of information items.
- Relevance.
- Related work: Information Extraction, Question Answering.

Indexing and the Boolean Model

- **Indexing**: Representing the documents from a collection in some type of **data structure** (called an **index**).
- For Boolean queries:
  - Term-Document Incidence Matrix (not scalable).
  - Inverted index: Postings lists.
  - Postings lists with skip pointers.
- Implementation of query operators: *AND, OR, NOT*
  - Query optimisation

Preprocessing

- Differences between a “word” and a “term”.
- Preprocessing steps & challenges:
  - Tokenisation
  - Stopword Removal
  - Stemming
  - Normalisation
  - Thesauri
  - Soundex
  - Phrase Queries
  - Positional Indexes

IR Models: Vector Space

- Model documents as a "bag of words" in *n*-dimensional space.
- Vector algebra used to calculate similarity between documents and queries: cosine similarity.
- Non-binary term weights (e.g. TF-IDF).
  - Better retrieval, as query terms are weighted according to influence on documents.
  - Strong term appears often in a document, but rare overall.
- Output: **ranked** list.
- Advantages: Improved performance, partial matching, ranked list.

IR Models: Probabilistic Model

- Based on the probability that a document is a member of the ideal answer set, *R*.
- Based on Bayes' rule, a formula is calculated that allows a similarity score to be calculated based on the following two probabilities for terms (where k_i is a term):
  a) The probability that a relevant document contains k_i
  b) The probability that a non-relevant document contains k_i
- Initially, have to guess reasonable values.
- Once user gives feedback on relevant docs, probabilities get closer to the real values.
- **Advantages:** ranked list, user feedback possible.
- **Disadvantages:** need to guess initial probabilities, only binary term weights.

IR Models: BM25

- Extension of probabilistic model to add weighting and remove need for user feedback.
- Based on three principles:
  - Inverse document frequency
  - Term frequency
  - Document length normalisation
- Variations: BM25F and BM25+
- Most researchers agree that it outperforms the vector space model on general collections.

Evaluation

- IR evaluation based on the *Cranfield paradigm*.
  - Test queries on a known document collection.
  - Human judges decide if docs are relevant.
  - Precision and Recall scores based on answer set and relevant set.
- Modern systems have challenges: ranked lists, incomplete judgments, degrees of relevance
- Newer metrics address some of these:
  - P@10, R-precision, MAP, bpref, NDCG

PageRank

- Method of measuring the importance of a page.
- Different to traditional publishing.
- High PageRank if:
  - Many backlinks
  - Backlinks with high PageRank
- Pages contribute to the PageRank of their outlinked pages: recalculated many times until the scores converge.
- Possibility of rank sinks means that a damping factor is required to avoid inifinite PageRank.

Web IR Challenges

- Finding information: how web crawlers work.
- Scale of the web.
- Adversarial IR
  - Meta tags, hidden text, cloaking, JavaScript, doorway pages, exploiting PageRank

Fusion

- Joining results from different systems together into one.
- Different applications based on level of database overlap:
  - Overlap level affects the type of algorithm we use.
- Effects that can be exploited: skimming, chorus, dark horse.
  - Algorithms exploit effects in different ways.
- Rank-based (interleaving, weighted interleaving, election type algorithms).
- Score-based (CombSUM, CombMNZ, Linear Combination)
  - Score normalisation
- Segment-based (ProbFuse, SegFuse, SlideFuse)

Final Exam

Final Exam

- 70% of final grade.
- Friday 16th June: 9:55am-11:55am (2 hours)
- 4 Parts.
  - Part 1 is compulsory (30 marks)
  - Choose any TWO other parts (35 marks each).
    - If you attempt to answer all 3 other parts, let me know which 2 you want me to grade (otherwise I will grade the first 2 I find).
- **Note:** the module material is not exactly the same as last year, so parts of last year's exam may not be familiar to you.

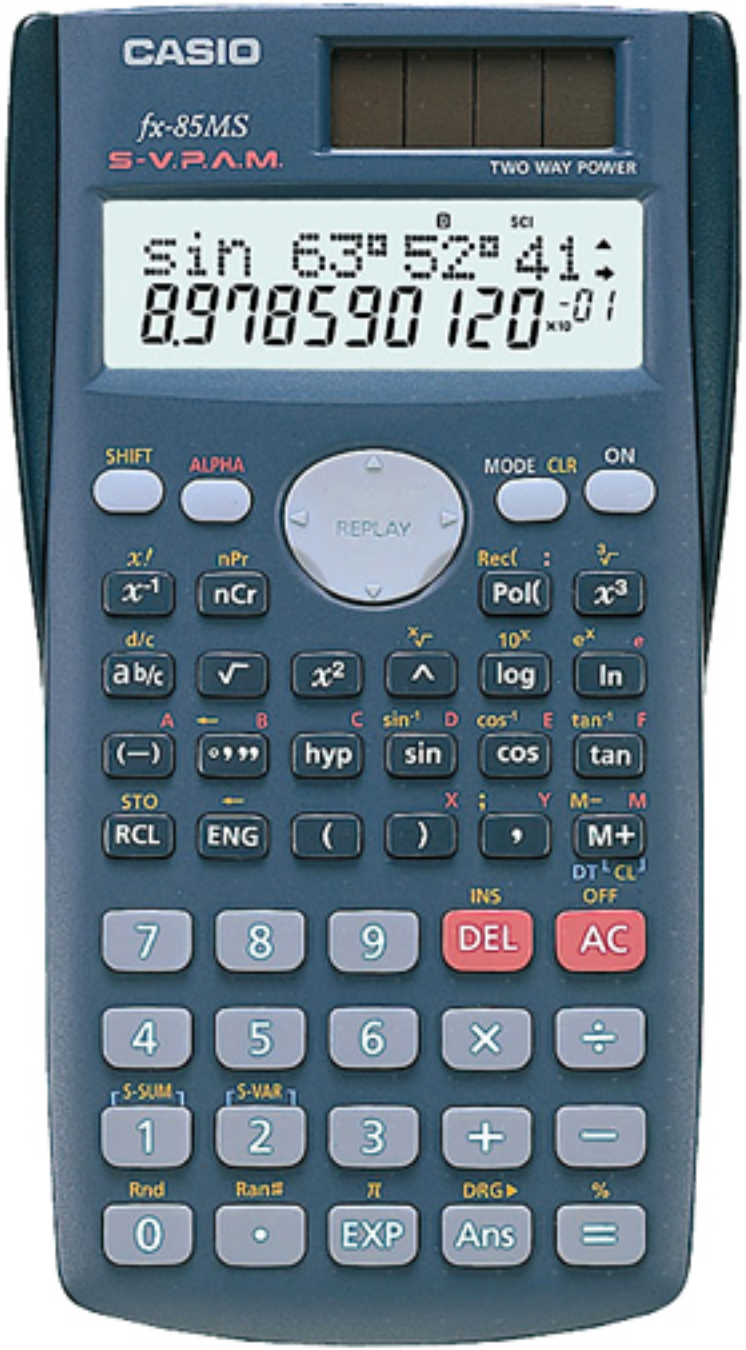
Calculations

The exam will involve some calculations.

You can bring a **non-programmable calculator** into the exam with you.

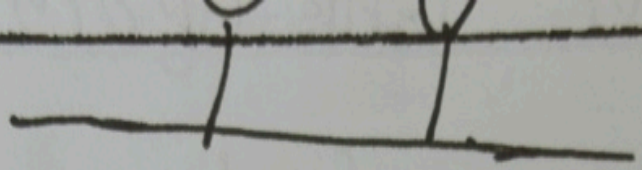
Phones are

NOT allowed.

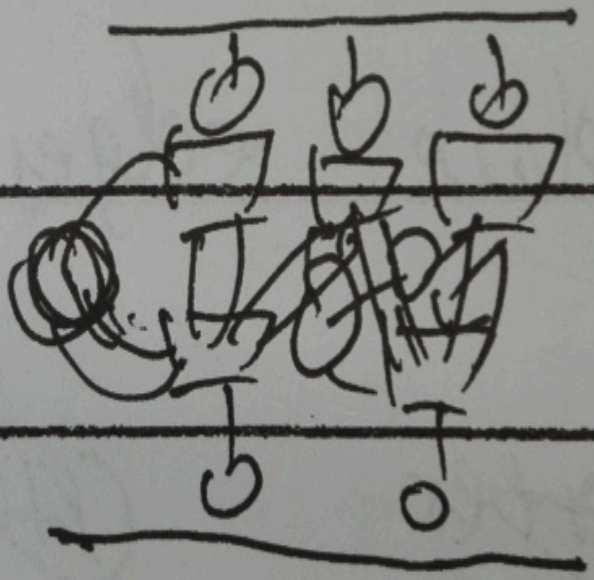


Exam Advice

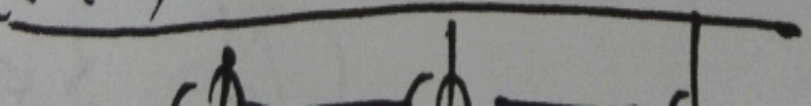
- The exam is for you to show me what **you know**.
  - **Analysis** and **understanding** is important: you're Stage 3 now!
  - Just being able to write the contents of PowerPoint slides is not enough to get a high grade.
- I can't give you marks for things you don't write.
  - It is better to write too much than to write too little.
  - If you want more paper, just ask for it.
- I can't give you marks for things I can't read!
  - Make sure your writing is neat and your diagrams are clear!
  - PLEASE, PLEASE, PLEASE, don't do what these students did:



Each vertex has more than three edges



Vertex



Question 1: a) A tree is a hierarchical ADT that supports relationships in terms of parent-child relationships. A node can have zero or more children. Every node has at most one parent. There is always one node in every tree that has no parent at all, and this is known as the 'root node'. Attached to the root node, you have the left subtree, and the right subtree. These are the children of the root node. These children themselves are the root nodes of the subtrees.

- remove(e) → removes node containing e
- insert(e) ← insert node containing e
- iterator() ← returns an iterator of the elements in the tree
- positions() → returns an iterator of the positions of the elements in the tree
- isEmpty() → tells you whether or not the tree is empty
- size() → no. of nodes
- isInternal(p) → has p got children?
- isExternal(p) → is p a leaf node?
- isRoot(p) → is p a root node?
- root() → returns the position of the root
- parent(p) → returns p's parent position
- children(p) → returns an iterator of p's children positions

2

10

Exam Advice

- If you have doubts about describing something in English, maybe a diagram/picture will help?

- **BUT:** a diagram should **ALWAYS** be explained: a diagram with no explanation is **USELESS**.

- Read the paper before answering any questions. I will be visit the exam room during the early part of the exam. If you do not understand what a question is asking, you can **ask me**.

Exam Advice

- Do not write on the exam paper: answer books will be provided.
- During the exam, do not talk to anybody or communicate with other students. If you need me or an invilator to come to you, raise your hand.
  - If you need to borrow something from another student, you **must** ask an invigilator to get it for you.
- Using phones in exams is banned, and is a serious offence.
- Also, this is a "closed book" exam: you may not bring any notes or study material with you.
- Good luck!