COMP90044 Research Methods

Assignment B: Research Paper Review (Masters of Philosophy and PhD Students)

Due: 11:59pm, 31 August, 2018
Mark component: 20% of assessment

Method of submission: Electronic copy via the LMS. Note that submissions will be processed using Turnitin. Any assignment submitted late will incur a penalty of 0.5 marks for every day or part of a day that it remains not submitted (both week days and week ends).

Task

1. Together with your supervisor, choose *one* research paper from a recent refereed conference [1]. This paper should be from the area you chose for your literature review and could be one of the papers that was used in the review if you wish. (Supervisors: Help your students identify a paper that has significant, clear failings, but which could be improved with further work.)

2. For your chosen paper, write rough notes *for your own use* that answer the following questions.

   (a) What are the researchers trying to find out?
   (b) Why is the research important?
   (c) What things were measured?
   (d) What were the results?
   (e) What do the authors conclude and to what do they attribute their findings?
   (f) Can you accept the findings as true?
   (g) Is the work appropriately linked to previous literature?

   The answers should be substantially your own writing, not quotes, paraphrases, or illustrations from the paper.

   Discuss any failings or shortcomings of the method used to support the findings. Justify your opinions as carefully as you can. As part of the answers to these questions you should briefly summarize the proposed method and the results achieved.

3. Based on your rough notes, write a review of the paper, *including a clear recommendation as to whether to accept or reject.* Take care to discuss all of the major problems in the paper. Expect to address aspects such as:

   - Significance – is the work substantial or trivial?
   - Novelty – is it new, obvious, or unoriginal?

[1] Mechanical/Infrastructure Engineering students - a journal paper may likely be more suitable for your discipline. Please discuss with your supervisor

- Soundness –is the work to a good standard, and well defended against reasonable criticism and skepticism?
- Clarity – can it be understood?

Expect to write 1000–1600 words for the review. Make use of resources about reviewing: lecture notes, the chapter on refereeing in the textbook, and guidelines you might discover on the web.

4. After writing your review, criticize and evaluate your chosen paper's structure and presentation using the following criteria and then revise your review appropriately.

   (a) Is the ordering reasonable (of sections and within sections)?
   (b) Are sections linked together?
   (c) Does the paper flow? Are important elements appropriately motivated and introduced?
   (d) Where is the literature review?
   (e) Is there an accessible abstract and introduction?
   (f) How carefully has the paper been edited?
   (g) Are there aspects of the presentation that could be improved?

5. By the due date above, submit through the LMS your review. Make sure that you mention at the start of the review the full bibliographic details of the paper being reviewed.

6. In completing this assignment, you are required to adhere to the University's standards regarding academic honesty and you must avoid plagiarism. Further information about what is expected can be found at
   `http://academichonesty.unimelb.edu.au/plagiarism.html`

Sample review: of a paper on Ternary Search Trees

Included as an example of the kinds of comments that might be made about a flawed but rescuable paper.

This paper concerns improvements to ternary search trees. Following through the idea that TSTs can be rebalanced to reduce access costs, the authors propose and test a variety of rebalancing schemes, including splay-like adaptivity and explicit cost-based balancing.

The application of rebalancing to TSTs is novel, I believe, and the authors have tried an interesting range of rebalancing techniques. The idea is worthy of publication, even if (as may be the case in light of the comments below) it is not as effective as other data structures. If nothing else, a careful exploration of these ideas will mean that other researchers don't re-invent the techniques.

In other words, in principle there is a lot to like about this paper, but it is not well executed. Many of the statements are arguable, key comparisons that are not favourable to the authors' thesis are passed over, there are serious questionmarks over the experiments, and the description and analysis of the algorithms and data structures does not meet the standards set in the other work in the area.

Working through the paper more or less in order, specific issues (some of them minor) are as follows:

In the abstract, the maths should be omitted – the second and third sentences contribute nothing to the reader's understanding. In the keywords, "Information Retrieval" is misleading and should be removed.

The first paragraph of the introduction doesn't motivate the work well. Phrases such as "totally ordered" (always true of strings) are a distraction. More significantly, the assumption of time-invariant probabilities undermines some of the later claims. With time-invariant probabilities, adaptation is relatively uninteresting – one of its great strengths is the ability to respond to local changes in data that don't correspond to global properties.

On page 2, it seems odd that the advantages over hashing are enumerated, but that the existence of advantages over BSTs and tries is merely asserted. The first two claimed advantages over hashing are debatable; the first does not make sense, and the second clearly depends on properties such as the load factor of the hash table.

This page introduces the bitstack trie as a point of comparison, and correctly identifies some bitstack-trie shortcomings, such as the need for parameters. (However, I do not know what the evidence is for the claim that the bitstack trie is inefficient for long strings; this is not supported by the Heinz-Badumen paper.)

The Heinz-Badumen paper is a useful point of reference because they also explore TSTs, hash tables, and BSTs, comparing them in a common experimental framework. It is surprising, to the point of being deceptive, that Badr and Oommen fail to mention that Heinz and Badumen found TSTs to be significantly slower and less compact than hash tables or bitstack tries. Worse, the implicit contrast drawn on the next page, where the advantages of self-adjusting structures are listed, suggests the reverse conclusion. There need to be explicit statements about the actual performance of TSTs compared to other, competitive structures.

The advantages of self-adjusting structures listed on page 3 need to be qualified. 1. This statement is significantly misleading. It is true in general that adaptive structures have these advantages, but in the general case (for example splay trees) there is a reduction in worst-case complexity, say from O(n*n) to O(nlogn) total cost. But in this case the possible savings are a constant factor, bounded above by the alphabet size – potentially substantial savings, but not of the same kind as available in the other cases. 2. Adaptive structures only require less space than cost-balanced structures, not than other structures in general. Because parent pointers are needed, they require more space than unbalanced structures. 3. This statement is arguable, as anyone who has attempted to teach adaptive structures to undergraduate students can attest.

Starting on page 11, the algorithms (pseudocode) are impossible to follow. I see no point in including this material. The splay algorithm on page 12 is particularly poor. A Knuth-style presentation, used in many of the papers cited, would be much superior for this material.

On page 13, the random approach needs motivation. Why is it worth considering? What function do the random values serve? In other algorithms of this kind, the distribution from which the values are generated is changed over time (to prevent nodes that are presumably rare from having a high likelihood of promotion).

On page 15, I presume WPL stands for weighted path length. It would be nice to have this defined somewhere.

It is curious that the theorems concern properties that are relatively minor; theorem 1 is more of an observation than a theorem. Why are the properties of the splayed and random variants of rebalancing not treated in the same way? Perhaps there is a good reason for this inconsistency, but it needs to be explained. I don't understand the statement of theorem 3; what is the significance of phi? What (in words, not maths) does the theorem mean?

The experiments, beginning on page 19, are highly inadequate. The formal arguments made for

the structures are weak (nothing in the analysis indicates whether the proposed structures are likely to offer advantages), so the whole argument rests on the experiments. And they are not persuasive.

Looking at them in detail, is there some underlying problems in the implementation? On a Pentium III of 700 MHz, Heinz and Badumen report that a TST can search, with insertions, 179 million strings in 95 seconds. Badr and Oommen report almost 400 seconds for 150 million accesses (figure 8). This is a large discrepancy.

Note too that the reported improvements are less than 20fastest method reported is still slower than a bitstack trie, and presumably requires substantially more space. (Heinz and Badumen reported bitstack tries to be smaller than TSTs, and the modified TSTs have additional fields. They reported about 70 seconds for the bitstack trie.)

Badr and Oommen do not report the space required for their structures. Nor do they compare to any other data structure.

Artificial data is used for the experiments. First, this is a surprising decision, given the large, standard collections available online. The largest real set considered is Moby Dick – a tiny collection of words by current standards. The complete text would fit in the cache of a current processor, so it does not present a challenging problem for data management.

Second, the artificial sets are not well explained. Strings vary in both length and frequency; in text collections such as Moby Dick length and frequency are inversely correlated; and there are distributions amongst the characters, which vary with character position. Third, there is no unique Zipf distribution (the power is a parameter). Indeed, all of the listed distributions have unspecified parameters. Fourth, what is the wedge distribution?

These experiments do not make a case for their structures, or for TSTs in general. They need to be designed and reported with much greater care.

An overall question that should be addressed is identification of a specific application where TSTs are clearly superior to the alternatives. Is it long strings, or sparse strings, or something else? The concrete examples used are instances where other structures have been shown to be better than TSTs.

Another overall question is the pertinence of some of the references, in particular the papers concerning TSTs. There may well be older papers in which it has been argued that TSTs outperform hash tables, but are these papers still relevant? The growth in cache size and CPU speed has led to development of cache-aware algorithms, of which a crucial feature is that the number of random memory accesses must be kept small. Compared to tries and hash tables, TSTs are poor in this respect. Cache issues should be considered in this paper.

Sample review: of a paper on a 'novel' index structure

Included as an example of the kinds of comments that might be made about a paper that cannot be rescued. Some reviewers might be more blunt, others more constructive.

The intended contribution in this paper has two principal elements. The proposals are based on the assumption that effective document retrieval can be based solely on a limited number author-provided keywords. The elements are a scheme for weighting the keywords and an index data structure that is intended to provide fast mapping from keywords to documents.

None of the proposals in this paper are of value. There are many problems in this work, including:
– Keywords are not an effective retrieval mechanism.
– The weighting system is based on invalid assumptions.

– The data structure is unscalable (and not particularly novel).
– The experiments are methodologically flawed, and do not support the conclusions.
– A great deal of relevant literature has not been considered.

Considering these issues in detail:

Much of the research in information retrieval and similarity computation is concerned with the need for full-text retrieval, with good reason: retrieval based solely on summary information such as metadata, titles, and keywords is ineffective. There are many reasons that keywords fails in retrieval (including vocabulary mismatch, reader interest in minor elements of documents, author failure to use a controlled vocabulary, and lack of vocabulary coverage), and there are many experiments that have shown that all words in text need to be indexed. The authors have cited a range of textbooks; they should be familiar with such basics of information retrieval.

Indeed, the failure of keywords as a retrieval mechanism is all too obvious to even casual users of text databases. I sugggest that the authors visit a repository such as the ACM portal and use the advanced search facilities to try to find documents using keywords or ACM categories only. Even author names and titles are of at best limited help. It is difficult to understand why the authors have chosen to develop a new approach to a form of searching that in practice is not useful. On this topic, it is curious that at no point do the authors discuss the scale of search that is required for typical text repositories – for example, at about 100,000 documents, the ACM digital library is one of the smaller collections of research papers.

The proposals in this paper are invalidated if keyword search is not useful. However, they deserve independent discussion, to explain to the authors why they would not be a valid contribution even if keyword search was of value.

First, the weighting system for keywords. This aspect of the paper is mystifying – the authors have not motivated the work clearly and do not relate their proposals to the many existing schemes for term weighting. Questions that are unanswered include: Why should keywords be differentially weighted? Why these particular weights? Why are keyword frequency (or IDF) and other such considerations not of relevance? Who are the experts who did the weighting? How does someone become an expert in weighting? (Studies have found that humans are not good at predicting the weights that lead to the most effective retrieval.) How might the weights be validated? There are any number of weighting algorithms in the research literature; what distinguishes this scheme from others?

There is a vast literature on the topic of weighting, but it has not been referenced. The authors know that this literature exists (they have cited several textbooks), but they seem unfamiliar with the arguments required to support a weighting scheme. They should consult Willett and Sparck Jones's "Readings In Information Retrieval" and landmark work such as the the recent pair of Robertson, Sparck Jones, and Walker papers in "Information Processing & Management" (2001).

Second, the index term structure. It does not meet the basic criteria required of a solution to this task. The construction algorithm is O(n.n), as addition of a node requires inspection of all existing nodes – practical indexing algorithms are approximately O(n) – and is ill-defined (the rules are incomplete, inexhaustive, and under-specified). The structure itself cannot scale. As documents are added, search is increasingly exhaustive, and, under a vocabulary that is likely to be limited, clusters become less distinct. Even in the small example shown in Figure 2, it seems that identifying a document requires undirected traversal of irrelevant nodes before all relevant nodes are discovered. There is no consistent way to explore the structure – a problem that will lead to significant costs when the number of documents is large.

The claims for the structure are misleading. In what sense does it provide document management? In what sense do terms inherit? Very rarely is the set of keywords for one document a subset of the terms for another; and the documents do not inherit from each other.

Crucially, the structure has not been compared to any relevant existing structure. An inverted file is not a valid point of comparison. Inverted files are designed to provide fast query evaluation when the index is large (often many times the size of the available memory); they are designed to be disk-resident, although can be effective when used in memory. In contrast, the TIS could never be used on disk, due to the use of pointer chains – the disk accesses required to get from one node to another would make it impractically slow. I expect, as argued above, that the TIS will for ranked queries dramatically slow as the number of documents is increased. Nor is the inverted file implementation in MG a good benchmark, as compression is used to optimize use of memory and communication channels. If the aim is produce a method that is fast and operates over small volumes of data only, compression is inappropriate. It is probably the use of compression in MG that accounts for the relatively slow performance of inverted files. This comparison, between MG's version of inverted files and TIS, is about as sensible as comparing a new car to a submarine; the car may be faster but the two have no design objectives in common and the comparison is meaningless.

The appropriate point of comparison would be an in-memory structure such as a K-D tree or a trie. The TIS is in fact a form of trie, where each level represents inclusion of an additional index term. No comparison of any sort to any in-memory structures has been made.

Problems with the experiments are not limited to use of an inappropriate benchmark. Evaluation of information retrieval systems has a long history, with many papers and texts discussing approaches to system measurement. In this paper, standard evaluation methods are entirely ignored. There is no discussion of recall and precision or of the needs of searchers; the figures quoting near-100% accuracy are ludicrous. (Of course a Boolean system can accurately search for things that match a condition, but that does not mean that they are valid answers to a user's query.) This total neglect of the principles of information retrieval poses strong questions about the authors' understanding of the area.

The experiments also have practical shortcomings. Ranking is discussed (implicitly), but the experiments evaluate the use of the structure for Boolean querying, not ranked querying. (It is not clear whether the runs with MG take the same input and produce the same output, but it seems unlikely, further invalidating the comparison.) The efficiency measurements are on a small scale; how confident are the authors that these measurements will scale up? The data sets used are so small that they can comfortably reside in the CPU cache of a small desktop computer, so that pointer access costs are a tiny fraction of that for larger data volumes. Also, this data was artificially generated – a strange decision given the vast volumes of data that are available for such experiments, both online and in the standard test collections. It seems unlikely that randomly-generated data will have realistic characteristics. Speeds are reported in microseconds, which cannot be measured on a typical computer.

The data set used for effectiveness experiments is so small (less than 200 documents) that the results are statistically meaningless.

There are other problems with the paper, not major but still significant. In light of the deep flaws discussed above, listing of these is perhaps unnecessary, but it is concerning that the paper includes so many errors and misconceptions. Some examples:

Page 3, What makes the vector product "conventional"? What is the evidence that vector products are "inappropriate" or that they are inferior to the authors' methods? Their experiments do not measure vector products and the claim is not substantiated.

Page 5, The description of inverted files is inaccurate. The "table" does not "link to" documents and term lists are not logically combined; they are statistically combined. Query evaluation can proceed in many ways, not just term order.

Page 7, The citations for vector models and probabilistic models are absurdly out of date. The TREC and SIGIR proceedings include dozens of papers from the 1990s in which these approaches are compared.

There are many more examples of this kind.

The authors appear to have taken a great deal of trouble with this work but it is difficult to see how any of it can be rescued or repositioned as a worthwhile contribution. Had previous research been investigated more thoroughly at an early stage (so that the authors might have discovered, for example, that keyword-based retrieval is ineffective), this effort might have been directed more productively.