Dr. Gylfi Þór Guðmundsson
Adjunk professor
School of Computer Science
http://www.ru.is/starfsfolk/gylfig
Reykjavik University, Iceland

Reykjavik, 14. desember 2019

Review of Doctoral Thesis: Content-based exploration of unstructured data, authored by RNDr. Premysl Cech

Overview of thesis

The thesis covers a wide range of issues in the context of multimedia search and retrieval. First, a solid foundation is given before the basics of multimedia browsing and exploration is discussed in Chapter 2. Scalability challenges are discussed in the context of searching millions or billions of high-dimensional feature vectors and how low-response time can be hard to achieve, as not to alienate the impatient users. The to facilitate search there is a need to index the data but then one has to strike a balance between expressive high- dimensional features and fighting *the cures of dimensionality*. The challenges of presenting visual information to the users is discussed and several visualization techniques are presented as well as evaluation techniques.

In Chapter 3 we find a in depth discussion on interactive search engines, especially the VIRET framework that the author partook in creating and successfully exhibiting in the annual *video browser showdown* competition. VIRET, or the VIRET tool, has done very well at this event in the past few years, demonstrating the quality of work that has been put into researching and building the system. Highlighted in this chapter is also that it is not sufficient to build a good framework for multimedia retrieval but that user experience and usability is equally important. This is highlighted in the lessons learnt from the 2019 competition with novice users where some results being found but not noticed by the inexperienced users.

Next, in Chapter 4, we have a discussion on how the methods we frequently apply to multimedia can find uses in other domains. This requires however to model the new domain on a vectorial structure and explained is how a vast amount of HTTPS network data was classified and made searchable. To cope with the scale of the data the Map-Reduce framework is used as it assists with automatically distributing the workload over a large number of computing nodes. This allows users of the system to search for similar communication snapshots and discover if the traffic being analyzed is potentially malicious.

The primary contribution of the thesis given in Chapter 5, namely the implementation of distributed similarity joins in both Hadoop Map-Reduce and Apache Spark. The motivation given is that the volume of data has outgrown the capability of single machines and thus new solutions, that utilize distributed and parallel computing, are needed. In addition, the chapter proposes approximate k-NN join algorithms that relax the stringent demand of the exact k-NN joins in an effort to gain significant improvement in speed. In an effort to devise effective and efficient k-NN similarity joins two older variations, and three newer algorithms, PGBJ, PAKJ and PEGKJ, are implemented and tested. PGBJ is a revised variant of the algorithm written by Lu et al. in 2012 that builds clusters by way of pre-selecting global pivots. PAKJ is a method based on relaxing the strategy of replicating whole Voronoi

cells using upper and lower bounds with a heuristic based approach. The indexing technique of M-Index is used and adds an upper bound maximum (MaxRecDepth) that limit the soft-assignment of each database object to pivots (clusters). This is done in an effort to curtail the issue that PGBJ runs into in high-dimensions where all the data is replicated to every pivot. Finally, PEGKJ is a further improvement that gives epsilon bounded guarantees on the approximation of the k-NN join.

Comments

While the theoretical aspect of the work is very good I find some of the practical aspect of the text to be lacking. In the the presentation of Map-Reduce in section 1.4.1. a figure is presented, Figure 1.14, that is all but useless for the reader to understand how Map-Reduce actually works. Here should have been presented the classical MR-pipeline of HDFS -> MAP -> Shuffle -> REDUCE -> HDFS. The presented image is simply not useful nor aids the reader in understanding the challenges of implementing algorithms on the M-R framework. In the discussion about M-R, there is also no mention of the key concept that it is built around, namely that the execution order of the data must not matter and that Map taks are spawned where the data resides, i.e. that the work is sent to the where the data resides instead of the classical work of bringing the data to the processing CPU. I could go on in some detail on what I find missing in this description of Map-Recuce but rather I point out that in the discussion we get a description of how YARN works, and I wonder why that matters more then how M-R actually get work done.

In 1.4.2. we find the description of Spark and again the author fails to mention one of the fundamental difference between M-R and Spark in that Spark is a Master-Slave setup where the execution flows from the Master to the slaves when specific operations are applied to the RDDs. But otherwise the discussion was good.

In 2.1 - Scalability challenges: I find the assertion that *"...single machine computational power is not increasing very much..."* to be a bit strong as we are seeing ever more computing cores and GPU boards being packed into a single machine. That may however not be keeping up with the growth of datasets but one can't claim the power of these machines is not increasing. Also, I find it a vast oversimplification to say that parallelism can be *"...easily achieved..."* via grouping machines into computing clusters.

The main primary contribution of the author is discussed in chapter 5 where the following is claimed: 1) *"Because data volumes are often too large to be processed on a single machine (especially for high-dimensional data), we focus on the distributed MapReduce environment Dean and Ghemawat [2008] running on Hadoop and Spark."* and that you perform a 2) *"Thorough and extensive performance evaluation on large data with different dimensionality (from 10 to 1000 dimensions) running on fully distributed Amazon clusters, with most experiments evaluated on the Spark platform processing up to tens of millions of objects."*. When I look at the reported results I note that the datasets, in the space saving format, is only a maximum of 5GiB while each of the computing nodes have 15GiB of RAM. As there are used between 5 and 20 nodes the computing cluster has 75 to 300GiB of RAM. There is no further reporting on the size of the dataset and thus I am left wondering why this could not have been run on a single machine and weather the scalability claims have actually been satisfied. Also, missing is any analysis what actually happens during the execution of the jobs, like analysis of I/O traffic, how much of the time was spent on CPU vs. time spent on reading data from remote servers, disk or over the network. One of the key contributions of the PEGKJ algorithm is to limit the replication of data and yet there is not analysis done of the distribution of data over the clusters of any of the algorithms and the only values reported are the precision, time and effective error. The definition of "effective error" is actually never given, but in 1.2.4. approximation error is defined.

In section 5.5.5., The effect of dataset size. In this section we find a larger dataset, 1.2 million db-objects and 15.3 million query-objects. The size of the full dataset is however not disclosed nor fully analyzed what added overhead this will have on launching and running the M-R or Spark jobs. Because

of the high replication factor, the amount of scanning will be significant (i.e. a large fraction of the data is touched during the k-NN search) and thus the dominating cost will indeed be the CPU power of doing the Euclidean distance calculations. This must be stated somewhere as non of the factors I have assumed above are stated in the experiments. In other CBIR systems, that use larger and deep index hierarchies and more aggressive approximations the cost of scanning is actually not the dominating cost factor unless the joins are truly massive.

Bottom line is that in a thesis, were we are not bound by the truly annoying page-limit, one would expect a deeper discussion on all the potential bottlenecks of working with large amounts of data in a distributed environment. I/O operations are definitely part of the challenge unless work is done to prove otherwise. To avoid misunderstanding I am not saying that the work was not done in this case. I am merely suggesting that the reporting of said work is not properly demonstrated in this thesis and that, I think, is unfortunate.

Questions

1. Are all the algorithms you evaluated CPU bound during the whole running time of each job?
2. For each query, how much of the database was touched on average? And is this number linearly correlated with the running time of the jobs?
3. If the answer to question 1 was no, could you elaborate on how I/O costs, both reading the data from disk and network traffic during the shuffle process, factor into the overall running time of the k-NN joins?
4. RDDs in Spark support join operations. Would it not make sens to implement your algorithm for k-NN joins as part of the code for the RDDs in Spark?

Summary

From the list of publications, as well as the contribution chapters in this thesis it is clear that the candidate has been involved in most aspects of researching and building state-of-the-art multimedia search and browsing applications. He has chosen to tackle a very hard problem in trying to break the curse-of-dimensionality.

He has shown that he possess the the ability to acquire a deep understanding of the challenges and conduct meaningful and high quality research in addressing the yet unsolved problems. Equally important however is the skill of conveying your knowledge back to the community in a convincing and meaningful way. This is where the candidate still may require some practice.

It is my recommendation that the Faculty of Mathematics and Physics at Charles University accept this thesis and grant Premysl Cech a PhD degree.

Sincerely,

_____
Dr. Gylfi Þór Guðmundsson
Adjunk professor at the School of Computer Science
http://www.ru.is/starfsfolk/gylfig
Reykjavik University, Iceland