Exploring Complexity in Health: An Interdisciplinary Systems Approach
A. Hoerbst et al. (Eds.)

doi:10.3233/978-1-61499-678-1-582

MapReduce in the Cloud: A Use Case Study for Efficient Co-Occurrence Processing of MEDLINE Annotations with MeSH

Markus KREUZTHALER[1], Jose Antonio MIÑARRO-GIMÉNEZ and Stefan SCHULZ
Institute for Medical Informatics, Statistics and Documentation, Medical University of Graz, Austria

Abstract. Big data resources are difficult to process without a scaled hardware environment that is specifically adapted to the problem. The emergence of flexible cloud-based virtualization techniques promises solutions to this problem. This paper demonstrates how a billion of lines can be processed in a reasonable amount of time in a cloud-based environment. Our use case addresses the accumulation of concept co-occurrence data in MEDLINE annotation as a series of MapReduce jobs, which can be scaled and executed in the cloud. Besides showing an efficient way solving this problem, we generated an additional resource for the scientific community to be used for advanced text mining approaches.

Keywords. Big data, Cloud based processing, MEDLINE, Concept co-occurrence

1. Introduction

In the field of biomedical sciences, millions of papers are published every year, and part of this is stored in the literature database MEDLINE. Retrieval interfaces like PubMed support search scenarios that target the textual content of abstracts as well as the index terms added by specialists at the U.S. National Library of Medicine. These semantic annotations, using descriptors (concepts) of the multi-hierarchical annotation vocabulary MeSH form a rich knowledge base. To leverage this knowledge it is particularly interesting to analyze co-occurrence of descriptors across large amounts of MEDLINE content. MeSH descriptor co-occurrences at the level of MEDLINE records are periodically published in the co-occurrence file MRCOC [1] as part of the Unified Medical Language System (UMLS) Metathesaurus [2]. In the following, we will demonstrate a workflow that uses big data technology to process MRCOC in an efficient way. The goal is to compute significance values for MeSH descriptor co-occurrences from more than 1 billion records.

[1] Corresponding Author: Markus KREUZTHALER, Institute for Medical Informatics, Statistics and Documentation, Medical University of Graz, Auenbruggerplatz 2, 8036 Graz, Austria, E-Mail: markus.kreuzthaler@medunigraz.at

2. Methods

This task is well suited to be addressed by the MapReduce [3] programming paradigm for generating significance parameters of co-occurring MeSH descriptor pairs. The more the co-occurrence value of a given pair of MeSH descriptors deviates from random co-occurrence, the more relevant we can assume a meaningful association between these two concepts. For instance, a significant co-occurrence of a substance concept and a disease concept could be interpreted that this substance treats, prevents, or causes the disease.
MapReduce uses input key/value pairs, reports output key/value pairs, and can therefore be divided into two consecutive main tasks, *viz.* a *Map* and a *Reduce* step. By exploiting the *Map* functionality, the user accesses the input key/value pairs and generates intermediate key/value pairs. This intermediate pairs are handled by the MapReduce framework in use, which populates the key and the set of corresponding values to this key. In the *Reduce* phase, the user handles the logic of generating the output key/value pairs according to the populated key, accessing its values. In order to achieve the needed output format, we define six MapReduce steps, which are subsequently processed as six particular JobFlowSteps in a JobFlow, executed on Amazon Elastic MapReduce [4], with a managed Hadoop framework. Apache Hadoop [5, 6] is a Java-based MapReduce implementation for distributed environments and big data processing. The advantage is that the MapReduce task can be scaled up easily and parallelized running up to 20 parallel instances if needed, after having the problem modelled in adjacent MapReduce jobs. In the following, we describe how to solve this problem as a series of JobFlowSteps.

2.1. Log-likelihood ratio

The log-likelihood ratio [7] can be used for applying a chi-square test and is calculated via a co-occurrence table described in Fig. 1.

Co-occurrence	CUI1	¬CUI1
CUI2	#CUI1_CUI2	#¬CUI1_CUI2
¬CUI2	#CUI1_¬CUI2	#¬CUI1_¬CUI2

$$H = -\sum_{i}^{n} p_i(log_b p_i)$$

LLR = 2 (H(matrix) - H(rows) - H(cols))

H(matrix)	Matrix entropy
H(rows)	Sum of row entropies
H(cols)	Sum of column entropies

Figure 1. Log-likelihood calculation.

In our experiment, we are using the implementation from the Apache Mahout package [8] for log-likelihood calculation.

2.2. Step descriptions

Initial filtering and accumulation

Step 1: The first step accesses the detailed_CoOccurs_YYYY.txt, which describes the co-occurring MeSH descriptors at MEDLINE record level [1]. The file is compressed and uploaded in an Amazon S3 bucket, a storage drive in the cloud. This step populates the intermediate key/value pairs having the UMLS unique identifier (CUI) from MeSH descriptor one (CUI1) together with the UMLS CUI from MeSH descriptor two (CUI2) as key and the qualifier abbreviation from MeSH descriptor 2, which keeps the MeSH subheading information [9] as value. MeSH subheadings like DT = "drug therapy", TU = "therapeutic use" are important to disambiguate significantly co-occurring MeSH descriptor pairs, such as between a substance treating or causing a disease. A second co-occurrence is extracted in the opposite direction, so that for each record two intermediate key/value pairs are fed into Hadoop. In the *Reduce* phase, we are generating the final output line per key: CUI1 CUI2 [accumulated subheading information] #CUI1CUI2. CUI2 CUI1 [accumulated subheading information] #CUI2CUI1.

Intermediate occurrence calculations

Step 2: Takes the output from Step 1 and calculates the overall number of all MeSH descriptor occurrences. **Step 3**: Takes the output from Step 1 and calculates the absolute occurrences for CUI1 denoted as #CUI1. **Step 4**: Takes the output from Step 1 and calculates the absolute number of occurrences for CUI2 denoted as #CUI2. **Step 5**: The output from Step 1 is further extended by #CUI1notCUI2 using the information from the steps before, applying a reduce-side join, yielding: CUI1 CUI2 [accumulated subheading information] #CUI1CUI2 #CUI1notCUI2.

Final log-likelihood calculation

Step 6: The final step generates the intended output using the output from Step 5, again applying a reduce-side join with an additional log-likelihood calculation exploiting the currently available counts, which are needed for this step (#CUI1CUI2 #CUI1notCUI2 #notCUI1CUI2 #notCUI1notCUI2). This has the big advantage that per co-occurrence their relevance can be asserted using a chi-square test (with one degree of freedom) for different significance levels. The final output has the following format: CUI1 CUI2 [accumulated subheading information] #CUI1CUI2 #CUI1notCUI2 #notCUI1CUI2 #notCUI1notCUI2 log-likelihood.

3. Results

Experimental setup: Amazon instance information: Name: M1 General Purpose Medium; API Name: m1.medium; Memory: 3.75 GB; Compute Units (ECU): 2 units; Cores: 1 core; Storage: 410 GB; Arch: 32/64 bit.
Computing environment: Amazon Elastic MapReduce Managed Hadoop Framework in combination with Amazon S3 cloud storage.
Filter constraints: Co-occurrences within 5 years (e.g. time frame MEDLINE 2009-2013). Both MeSH descriptors are major topics (FLAG ZY). Benchmarks: see Table 1.

Table 1. Processing time (minutes) depending on the number of instances and calculation step. IFAA = Initial filtering and accumulation; IMOC = Intermediate occurrence calculations; FLLC = Final log-likelihood calculation.

Slave Instances	Calculation Part			SUM
	IFAA	IMOC	FLLC	
2	50	27	36	113
4	29	13	17	59
10	16	9	7	32

4. Discussion

The results in Table 1 demonstrate the feasibility of processing the MRCOC file for the use case presented in this paper in about 30 to 120 minutes depending on the number of slave instances in use. The described task was not effectively calculable using a single desktop machine (Win 7 Pro; Processor Intel(R) Core(TM) i5-3470 CPU @ 3.20GHz, 4 Cores; 8GB RAM) without parallel processing attached. By modelling the problem as a series of map/reduce jobs the task can be scaled and executed in parallel on a Hadoop cluster in a reasonable amount of time. The Amazon Elastic Map Reduce service was used for this purpose for negligible costs (a few dollars). This shows that batch job oriented reservation of virtual images in the cloud is easily available for users without their own clusters for big data processing. With the additional added log-likelihood information, statistical significance statements can be made regarding co-occurring MeSH descriptors. This allows for example disease-disease, disease-findings, or disease-substance stratifications. The vector of subheading information is relevant for guessing the correct predication between MeSH descriptors of certain semantic types, as exemplified by Table 2.

Table 2. Examples of semantic relations between concepts ordered by semantic types.

	Disease	Finding	Substance	Organism
Finding	sign of symptom of	accompanied by	treated by	affects caused by
Substance	causes treats prevents metabolite	causes treats prevents	interacts with	affects is produced by
Organism	causes affected by	causes observed in organism	sensitive to	interacts with
Body part	possible location of	possible location of	targeted by	targeted by

We had used this additional data on initial clustering experiments with promising results [10, 11].

5. Conclusion and Outlook

This study examined effective processing of big co-occurrence data in a cloud-based environment. As use case, log-likelihood calculation in combination with accumulated sub-heading information from co-occurring MeSH descriptors was used. Processing of

more than 1 billion records was feasible in less than one hour with a moderate setting of a Hadoop-based cluster in an inexpensive Amazon Cloud environment.
In the near future, we want to make this resource available to the public and see this as an additional format of the UMLS MRCOC file. We also want to switch from our Hadoop based implementation towards Apache Spark [12], as it seems that this environment has advantages in the sense of code readability and processing speed. An extension of this method is planned to propagate the co-occurrence data vertically across the MeSH hierarchy, which might result in 2 – 3 orders of magnitude larger input files.

Acknowledgement

This paper was performed as a part of the BMFacts project (BMFacts: Knowledge acquisition for a biomedical fact repository), funded by the Austrian Science Fund (FWF): [M 1729-N15].

References

[1] UMLS MRCOC file: https://mbr.nlm.nih.gov/MRCOC.shtml, last access: 13 Mar 2016
[2] O. Bodenreider (2004). The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic acids research*, 32(Suppl 1), D267-D270.
[3] J. Dean & S. Ghemawat (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1), 107-113.
[4] https://aws.amazon.com/elasticmapreduce/?nc1=h_ls , last access: 13 Mar 2016
[5] M. Bhandarkar (2010). MapReduce programming with Apache Hadoop. *2010 IEEE International Symposium on Parallel & Distributed Processing* (IPDPS), (pp. 1-1). IEEE.
[6] R. C. Taylor (2010). An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *BMC Bioinformatics*, 11 (Suppl 12), S1.
[7] T. Dunning, (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1), 61-74.
[8] R. Anil, T. Dunning, & E. Friedman. (2011). *Mahout in Action* (pp. 145-183). Shelter Island: Manning.
[9] Medical Subject Headings https://www.nlm.nih.gov/MeSH/subhierarchy.html, last access: 13 Mar 2016
[10] J. A.Miñarro-Giménez, M. Kreuzthaler, & S. Schulz (2015). Knowledge Extraction from MEDLINE by Combining Clustering with Natural Language Processing. *AMIA Annual Symposium Proceedings* (Vol. 2015, p. 915).
[11] J. A. Miñarro-Giménez, M. Kreuzthaler, J. Bernhardt-Melischnig, C. Martínez-Costa, & S. Schulz. (2014). Acquiring Plausible Predications from MEDLINE by Clustering MeSH Annotations. *Studies in Health Technology and Informatics*, 216, 716-720.
[12] A. G. & T. R. Soomro (2015). Big Data Analysis: Apache Spark Perspective. *Global Journal of Computer Science and Technology*, 15(1).