Scaling Archived Social Media Data Analysis using a Hadoop Cloud

Javier Conejero
Dept. of Computing Systems
University of Castilla-La Mancha
Albacete, Spain
Francisco.Conejero@uclm.es

Peter Burnap, Omer Rana
School of Computer Science & Informatics
Cardiff University
Cardiff, UK
{p.burnap;o.f.rana}@cs.cardiff.ac.uk

Jeffrey Morgan
Cardiff School of Social Sciences
Cardiff University
Cardiff, UK
MorganJ51@cardiff.ac.uk

***Abstract*—Over recent years, there has been an emerging interest in supporting social media analysis for marketing, opinion analysis and understanding community cohesion. Social media data conforms to many of the categorisations attributed to "big-data" – i.e. volume, velocity and variety. Generally analysis needs to be undertaken over large volumes of data in an efficient and timely manner. A variety of computational infrastructures have been reported to achieve this. We present the COSMOS platform supporting sentiment and tension analysis on Twitter data, and demonstrate how this platform can be scaled using the OpenNebula Cloud environment with Map/Reduce-based analysis using Hadoop. In particular, we describe the types of system configurations that would be most useful from a performance perspective – i.e. how virtual machines in the infrastructure should be distributed to reduce variability in the analysis performance. We demonstrate the approach using a data set consisting of several million Twitter messages, analysed over two types of Cloud infrastructure.**

***Keywords*-COSMOS; Twitter data analysis; Hadoop; OpenNebula Cloud;**

I. Introduction

The growing number of people using social media to communicate with their peers and document their personal everyday feelings and views is creating data on an epic scale [1]. Such data provides the opportunity for social scientists to conduct ethnographical research through discourse and content analysis of social interactions, providing an additional insight into modern 'online' society. However, the tools and methods required to conduct such analysis are often isolated and/or proprietary. The Cardiff Online Social Media Observatory (COSMOS) platform aims to provide mechanisms to capture, analyze and visualize data harvested from online repositories and feeds, in particular interactive and openly accessible social networking sites such as Twitter. The aim of the project is to translate our underlying social observation and analysis mechanisms into an embedded research tool that supports the development and execution of defensible academic social research methods for digital social data.

Cloud computing infrastructures can be used to host data analysis capability for social media data. A number of commercial companies, such as Radian 6 (now part of SalesForce.com) and Palantir utilize proprietary Cloud-based infrastructures (generally using Hadoop) to undertake such analysis. However, when such analysis is carried out over public Clouds – such as Amazon, RightScale, etc – users generally see a high variability in performance. Such variability can arise due to the number of virtual machines (VMs) hosted on a physical server, cross talk between VMs, the size of data being processed, etc. A number of studies have already highlighted such variability [2], [3]. Our focus in this work is to highlight how such variablility can be: (i) identified; (ii) exposed, to a user to enable them to make more informed resource selection – with a particular focus on the use of the Hadoop (MapReduce) analysis infrastructure.

COSMOS can support data ingest, analysis and visualisation of the results in real time. The benchmarking tool used for the data analysis is the sentiment analysis tool SentiStrength [4]. The reason for using this tool is that sentiment analysis, opinion mining and public 'moods' have received significant attention in the research community [5] – especially in relation to social media. We are particularly interested in how such experiments can be conducted at scale. Experiments have been successful on a single desktop computer with 4GB of RAM and a 2.83GHz Intel Core 2 Quad CPU. However, COSMOS also aims to support researchers in performing empirical longitudinal studies (e.g. public response and reaction to political agenda, changes in legislation, views on national identity etc.) and for this we need to process significantly larger volumes of archived data (for longer periods of time – ranging from 1 day to months or even years), for which terabytes of archived tweets (stored in plain text files) need to be processed. This is generally not viable on a single machine, so in order to handle this scenario we have developed a Hadoop application which exploits the Map/Reduce paradigm.

In this paper we describe the framework designed to run the application using virtualized Hadoop clusters in an OpenNebula [6] based Cloud environment; and present the results of a scalability analysis for the Hadoop application, using SentiStrength, on a range of data archive sizes and deployment strategies. We demonstrate how a Hadoop deployment over a single cluster vs. one that that is distributed across multiple machines can be supported, and the associated tradeoffs in choosing one over the other.

The paper is structured as follows. In Section II related work is presented. Section III provides a brief overview of the COSMOS system and describes the problem associated with archived data processing. The system implemented to support sentiment analysis at scale is described in Section IV, which is the main contribution of this article. Subsequently, experimental results are presented in Section V. The conclusions derived from the work and suggested future work is presented in Section VI.

II. Related Work

There have been several efforts in social media analysis for marketing, opinion analysis, understanding community cohesion, etc. There are some proposals that perform social media data analysis on–the–fly, such as Prometheus [7] and Truthy [8]. Prometheus is a P2P service that collects and manages social information from multiple sources in order to apply social inferencing functions, while Truthy is a web service that tracks political *memes* in Twitter and helps to detect astroturfing, smear campaigns, and other misinformation in the context of U.S. political elections.

Significant literature in social media analysis generally focuses on looking for bag-of-words, parts-of-speech or combinations of these (using an n-gram detection approach, for instance). Xiang et al. [9] identify how offensive content (containing profane language) can be automatically detected from a Twitter data stream, using a statistical topic modelling and machine learning approach. After collecting twitter data, they apply various pre-processing operations (such as removing re-tweets, URLs in tweets, etc) followed by inferring topic distributions from remaining tweets. The Hadoop framework is used to extract training tweets from a large tweet corpus, to enable lexical patterns to be deduced from the extracted tweets. They were able to demonstrate a *true positive* detection rate of between 70%-75%.

Twitter has also made use of Hadoop [10] for undertaking *predictive* data analysis on twitter feeds, using machine learning extensions made available in the data flow language Pig Latin [11] (which provides primitives for carrying out common operations on data, such as projection, join, group, etc). Scalability in their work is achieved by ensemble methods, whereby multiple machine learning algorithms can be executed independently and their outcome subsequently combined. The core focus of the work in the use of Hadoop in [10] has been the use of data analytics tools utilizing capabilities built into Pig.

Most of these approaches make use of Hadoop execution on *bare metal*, with very limited use through virtualised environments. Our focus is somewhat different and complementary to these approaches. In COSMOS we focus on understanding performance issues associated with data selection and sampling, and subsequently understanding issues of performance variability that arises when executing huge amounts of big data analysis using Hadoop on a Cloud environment that utilizes a virtualised infrastructure (exploiting its advantages, such as scalability, live migration, etc.). We believe this is much more representative of the type of work carried out in a research environment. In addition, we use Hadoop at different stages of the overall analysis pipeline – for both data selection and pre-processing and also to carry out sentiment analysis using SentiStrength.

Apache Hadoop [12], [13] is a framework for running applications on large clusters, providing reliability and support for transparent data migration. It provides the Map/Reduce paradigm, where the data is divided into smaller data chunks and distributed to the worker nodes where they are processed, and then the results are collected and combined. It also provides a non-POSIX compliant file system – the Hadoop Distributed File System (HDFS), to store data across the compute nodes. Moreover, it provides limited fault tolerance mechanisms by replicating data across multiple nodes in the resource pool.

Hadoop has become the *de-facto* standard in the research and industry use of the Map/Reduce paradigm across large-scale clusters. There are, however, alternatives, such as BashReduce [14], Disco [15] and Storm [15]. BashReduce is a Map/Reduce implementation that can be used as a Unix script using awk, grep, etc. It is less complex than Hadoop and lacks flexibility and fault-tolerance mechanism and, as a consequence, is less efficient. On the other hand, Disco is closer to Hadoop in terms of complexity and implementation. It is developed by Nokia Research Center and can be considered a good Hadoop competitor, but because it uses the operating system's file system and handlers, it also lacks fault tolerance on the associated storage. Hence, both BashReduce and Disco are less efficient than Hadoop.

Storm (also known as Real-Time Hadoop), on the other hand, is a distributed, fault-tolerant, real-time and open source computation system developed by Twitter. It is aimed at real time analysis using large clusters. Its objective is the same as Hadoop, but the usage mode differs, as Hadoop is meant to perform batch processing instead of real time processing. Both projects (Hadoop and Storm) are shifting towards big-data applications.

III. The COSMOS system

The Cardiff Social Media Observatory (COSMOS) aims to support social scientists in analysing socially significant data (e.g. tweets, blogs and news stories). The volume of data produced on a daily basis requires significant computational resources to analyse. For example, COSMOS collects around 3.5 million tweets a day. To perform a longitudinal analysis of, say public opinion and sentiment, around a socially significant event (e.g. a political campaign, change of legislation, world sporting event etc.) could require analysis of several weeks' worth of data. An example study may be public opinion surrounding the

London 2012 Olympics, where a study of opinion for two weeks before, during and after would require the analysis of 21 million tweets. On a single desktop computer this could take approximately 20 minutes. This is perfectly acceptable as a batch-processed computational exercise, but the reality is that social media analysis may require several "tweaks" to the study parameters, and therefore requires a more interactive way of analysing data. For example, age, gender, location and topic of study within the event may change, as hypotheses are formed and tested. Therefore, the computational analysis must be able to complete much faster to give a more acceptable wait-time for the researcher. The researcher should be able to invoke the computational resources to support large-scale data analysis on-demand and resources need to be dynamically allocated depending on the size of the job.

Over recent years, there has also been an abundance of social media analysis tools, created primarily to facilitate on-line advertising and marketing, from companies such as Radian 6, Palantir and IBM. Many of these tools however are difficult to use in the context of social sciences research due to their cost and proprietary closed (black box) technology that does not lend itself to methodological inspection. COSMOS proposes the use of a data model and a suite of openly available text analysis tools, which enable questions to be formulated that are relevant for social sciences researchers and for analyzing self-reported data [16]. Twitter provides a number of Application Programming Interfaces (APIs) that allow researchers to capture a percentage of the stream of messages being posted by its users [17]. At the 'firehose' level, researchers can capture every message that is posted, at the 'gardenhose' level they can get 10%, and at the 'spritzer' level 1% of the messages posted can be harvested. COSMOS currently harvests and archives the 'spritzer' stream using the Twitter Streaming API, and makes it available to researchers for inspection and analysis. Even at 1%, the API provides COSMOS with approximately 3.5 million message (or 'tweets') per day, which are processed in real time for some purposes and archived ($\sim$525 MB in plain text files).

A. *Data Archiving in COSMOS*

COSMOS captures tweets at a rate of approximately 30 to 100 tweets per second, where the structure of each tweet is described in [17]. The first COSMOS implementation stored information about each tweet and the Twitter user that authored the tweet in a normalised MySQL database using the 140Dev PHP library [18]. Although adding information about each tweet required a simple SQL insert at the end of the tweets table, updating the users table required an SQL select to determine if the Twitter user was already in the table and a subsequent SQL insert to add the user to the table. However, after accumulating approximately 10 million rows of Twitter user data, MySQL was unable to handle the load of approximately 30 to 100 select and insert operations per second on the users table on a single computer. To overcome the weakness of the relational data model for archiving Twitter data, we implemented a bespoke archiving solution that stores the information about tweets and Twitter users collected by the 140Dev library as serialised PHP objects in a hierarchical filing system. To improve search over Twitter data, we use a hierarchical filing system to provide a date-based index for archived data. Hence, data for each day is stored in a separate directory named after the day, month and year in which the data was captured. To focus our data set we extract from the serialised PHP objects a subset of tweet and Twitter user data that we store in flat-file CSV format.

In contrast to a normalised MySQL database, our current solution stores information about the author of each tweet with the tweet itself. This means we repeatedly store the same user information each time a Twitter user authors a tweet. We took our cue for this approach from Google's Bigtable [19] – a distributed storage system that stores copies of related data together for two reasons. First, data stored in close proximity reduces disk seek times which makes accessing the data faster. Second, storing all the data required to answer a query enables distributed retrieval solutions such as the use of the Map/Reduce algorithm. Figure 1 shows the various components that make up the COSMOS system and services within the system, such as "language detection", "gender detection", etc.

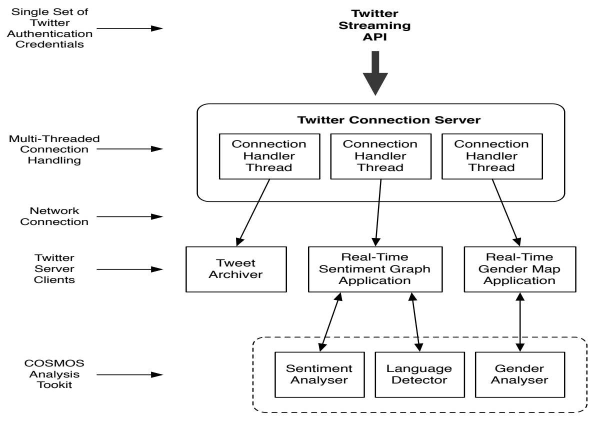
Single Set of Twitter Authentication Credentials
Twitter Streaming API
Twitter Connection Server
Multi-Threaded Connection Handling
Connection Handler Thread
Connection Handler Thread
Connection Handler Thread
Network Connection
Twitter Server Clients
Tweet Archiver
Real-Time Sentiment Graph Application
Real-Time Gender Map Application
COSMOS Analysis Tookit
Sentiment Analyser
Language Detector
Gender Analyser

Figure 1. Architecture of the COSMOS system

B. *Sentiment Analysis*

In our initial experiments with sentiment analysis, we used the Alchemy API [20]. Given a piece of text through a RESTful API call, Alchemy returns a floating point number between -1 and 1 that represents Alchemy's evaluation of the *emotional content* in the text ($>$ 0 = positive sentiment; $<$ 0 = negative sentiment; 0 = neutral). As an online API, Alchemy has the same two disadvantages for our work as other online APIs. First, the round-trip time of each networked API call lengthens the time taken to calculate the sentiment score of each text. At a rate of approximately 30 to 100 tweets per second, any realistic amount of network lag causes network-based text analysis to fall behind the Twitter stream. The second reason online APIs inhibit large-scale processing is rate limitation. Alchemy allows only 30,000 sentiment analysis API calls per day for approved academic users. To put this figure in context, a limit of 30,000 sentiment analysis calls per day would enable us to process the tweets we collect in ten minutes (at a rate of 50 tweets per second. To overcome these two disadvantages of Alchemy, we use SentiStrength [21] – which is available as a downloadable package that we have installed locally. These two characteristics mean we can analyse an unlimited number of tweets at no charge and with no network lag.

IV. Architecture

Our system makes use of a distributed processing approach based on Hadoop Map/Reduce paradigm [22] over an OpenNebula based Cloud computing infrastructure, as illustrated in Figure 2. It is based on the study performed in [23], but extended to a more complex Hadoop deployment and contextualisation. OpenNebula [6] is an open source project focused on the development of an industry standard solution for building and managing cloud infrastructures. As OpenNebula does not provide virtualization mechanisms, we use the KVM (Kernel-based Virtual Machine) hypervisor [24] to manage the virtualization within resources. It is an open source software for Linux that provides a full virtualization infrastructure (with support for hardware virtualization extensions) within a single machine. KVM primarily supports `x86` and `x86_64` architectures, and its development is supported by RedHat. KVM extends a standard Linux kernel and supports hardware virtualization by the use of a `/dev/kvm/` interface. OpenNebula manages KVM in order to deploy, monitor and control any virtual machine on each physical machine associated with the local Cloud infrastructure. Some alternatives are XEN [25] and VMWare [26]. KVM was chosen because it has been demonstrated to be successfully used alongside the OpenNebula system.

OpenNebula provides Infrastructure as a Service (IaaS) natively, but it also offers mechanisms and interfaces in order to develop new functionalities and to allow the integration of existing products within Cloud computing. Subsequently, there is a wide and rich ecosystem around OpenNebula (formed by tools, extensions and plugins) that enhance the functionality provided by OpenNebula, such as: complex schedulers, hypervisor support drivers, interface translators and language bindings, etc. On the other hand, it does not provide other kinds of service support mechanisms or extensions as alternatives such as Nimbus (which makes use of the Globus toolkit and associated services).

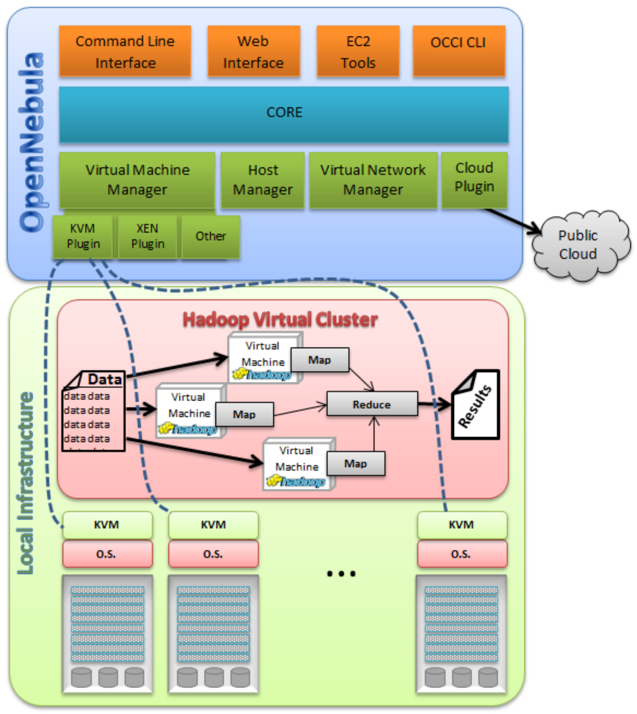

Figure 2. Data processing architecture for COSMOS using OpenNebula & Hadoop

Hadoop was deployed over a collection of virtual machines (VMs) making up a virtual cluster. OpenNebula allowed us to deploy as many Hadoop worker instances as needed, letting us decide the size and characteristics of the VMs/cluster and their deployment policy (dynamic contextualisation) depending on the amount of data to be processed. This also enabled us to migrate VMs to other Cloud environments. However, a computational overhead due to virtualization is introduced. Hadoop takes all the input files (from HDFS) and divides them into one line chunks, sending each one to a mapper. This action produces a massive number of mappers which can be spread over a large cluster. At the Map stage, each Tweet message is processed with SentiStrength in order to get the sentiment scores (negative and positive). Each result is generated independently, comprising the Tweet identifier and its associated positive and negative sentiment scores. With this design, the Map algorithm can be easily adapted to perform different analyses on individual Tweet messages by replacing SentiStrength with another analysis package.

```
map(String file_name, String line){
    String TweetId = getId(line);
    String Message = getMessage(line);
    String Scores = SentiStrength(Message);
    emit ((String) TweetId, (String) Scores);
}
```

The Reduce stage is not necessary due to the fact that the result combination is not needed for this application, but it may be necessary for other kinds of analysis. Currently, the Reduce stage outputs the results obtained by the Mappers.

```
reduce(String TweetId, String Scores){
    emit ((String) TweetId, (String) Scores);
}
```

The results obtained from the execution are stored into the HDFS, from where they can be consulted, acquired or even reprocessed with another Hadoop application. It is quite feasible that Hadoop may be invoked more than once during the analysis of a dataset to perform various types of analysis in sequence using a number of different executable packages.

The Cloud architecture (Figure 2) used in this work has been successfully tested over two local infrastructures, one at Cardiff University, UK and another at the University of Castilla-La Mancha (UCLM), Spain. The aim of having two different Cloud infrastructures was to investigate the behaviour of the application under different deployment policies.

A. Cloud Infrastructure

In this work we used Cloud infrastructure at the Universities of Cardiff and Castilla-La Mancha. The *Cardiff University* local infrastructure is composed of a cluster computer (Viglen ix4600) with one compute node and 2 Xeon e5620 CPUs (4 cores per processor + Hyperthread), 24 GB of main memory, and 4 TB of storage). It has CentOS 6.2 Linux [27]. The *University of Castilla-La Mancha* (UCLM) local infrastructure (known as Vesuvius) is composed of 10 compute nodes, with 2 Xeon e5462 CPU (4 cores per processor), 32 GB of main memory and 60 GB of storage each. It also has a headnode, with the same configuration but with 1TB of storage shared between all the compute nodes using NFS through a Gigabit Ethernet network. All of them have CentOS 6.2 Linux [27].

V. Experimental Results

In order to evaluate COSMOS with Hadoop, a virtual cluster is installed and configured. Each VM configuration (identifying the resources allocated to a VM), size of the cluster and deployment strategy defined for the experiments is described in the following subsection. Subsequently, the performance results obtained from the execution of the application for several million tweets are shown. We used JVM v1.7 (on both Cardiff and UCLM deployments) and Hadoop v0.20.0-cdh3u5 (from Cloudera).

A. Hadoop Virtual Clusters

The first step involves configuring the Hadoop virtual cluster (e.g.- number of Hadoop servers, number of workers and their VM parameters) on which the application is executed. We will show that different configurations have a direct impact on performance (i.e. the time taken to undertake the analysis). The Tweet analysis is performed with four different virtual cluster configurations (Table I). These 4 different configurations share 1 Hadoop server with 100GB storage, 6GB RAM, and 20% of the total CPU power allocated.

Table I
Hadoop virtual cluster (worker nodes) configurations

# Workers	CPU	RAM (Gb)	HDD (Gb)
1	70%	14	200
2	35%	7	100
4	17,5%	3,5	50
8	8,75%	1,75	25

These configurations are designed to maximize the usage of the resources of the Cardiff Cloud infrastructure (due to the fact that it has less main memory than UCLM Cloud infrastructure) and to ensure that they will work in both infrastructures. This is mandatory because OpenNebula 3.6 does not allow CPU and main memory overcommitment. Note that with these configurations we reserve at least 10% of the CPU power and 4GB of RAM for the OS and Cloud management tools within Cardiff local infrastructure.

For the experiments, each configuration is evaluated independently from the others, which involves stopping the workers that are not going to be used. The 4 different virtual cluster configurations are tested on the Cardiff Cloud, but only the 8 worker configuration on the UCLM Cloud, in order to evaluate the impact of multiple deployment strategies over the performance in a distributed cluster.

The aim of analysing multiple deployment policies is to determine if performance is perturbed when having multiple VM instances when the amount of CPU allocation is explicitly set. The objective being to understand how interference between VMs could result in an impact on performance of the overall application. The deployment of a more complex cluster with more workers would require smaller amounts of CPU dedicated to each one (due to CPU overcommitment limitation), so allocating less than $\sim$8% reduces their performance as it is not enough for each VM's OS and services to run efficiently.

B. Application Results

For the experimentation with the previously described application, a test tweet archive (of more than 2 Gigabytes), with up to 15 million tweets being processed to extract their sentiment. COSMOS currently harvests and archives the 'spritzer' stream using the Twitter Streaming API, and makes it available to researchers for inspection and analysis. Even at 1%, the API provides COSMOS with approximately 3.5 million message (or 'tweets') per day The tweet files are archived using a specialised hierarchical filing system, which stores tweets based on the day in which the collection was made. Hadoop also replicated these files multiple times in order to improve fault tolerance – this is achieved automatically.

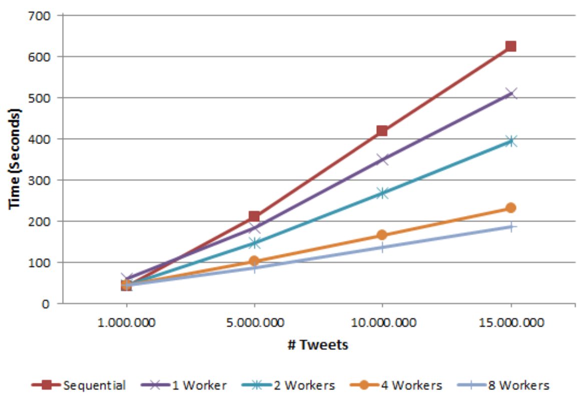

Figure 3. Cardiff Cloud Tweet processing performance

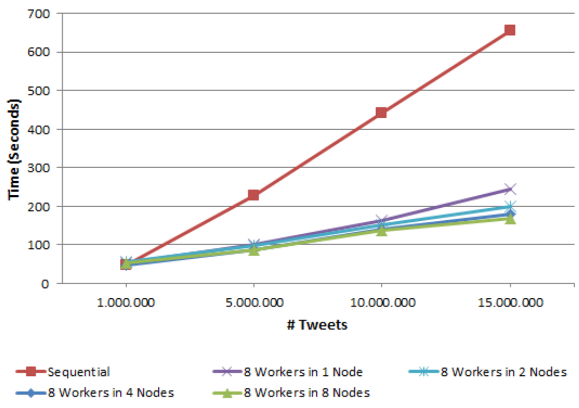

Figure 4. UCLM Cloud Tweet processing performance

The results obtained from the execution of the proposed application for the given tweet archive in the Cardiff Cloud (Figure 3) show an improvement on the performance (Table I) compared with the sequential version. It is shown that although the four configurations have the same amount of resources in common (but split between the total number of workers) within the same compute node, increasing the number of workers generally leads to an improvement in performance.

After observing the behaviour of the Cardiff Cloud with different worker configurations we focussed on the 8 worker configuration (as it shows better performance results) in order to experiment with the UCLM Cloud and evaluate if this deployment configuration, over a distributed cluster, affects performance. In this scenario the workers are distributed across the cluster and the results obtained illustrated in Figure 4. Workers are divided as 4 per node, 2 per node and then 1 per node – a node in this instance being a physical machine. Spreading the workers in this way does have an impact on the performance but not as much as with different worker configurations. The small difference in the 8 workers distributed across multiple nodes is due to each worker using one core on the node – with a total of 8 cores per node in the UCLM configuration. Hence, although 2, 4 or 8 workers are used on a node, each worker still makes use of a single core on the node. The main performance improvement in this instance is due to the high throughput resulting from having multiple workers operating on the twitter data. The slight improvement in performance between 2 works per node and 8 workers per node, for instance, is due to the high I/O per node when using 8 workers using the same network interface card. The UCLM configuration makes use of an NFS-based storage system. The best results are obtained when there is one worker per compute node, due to reduced I/O contention on the network interface card (although with reduced utilization of the computational capacity of a node).

Comparing the results from both Cloud infrastructures it can also be observed that for the best scenario (8 workers deployed across 8 nodes), the behaviours are quite similar, independently of where they are deployed. However, comparing the performance with the 8 workers configuration deployed in the same compute node at UCLM Cloud with the result obtained from the same configuration at Cardiff Cloud, it can be stated that Cardiff Cloud achieves slightly better performance. This is due to its compute node being more powerful than a UCLM Cloud compute node. This performance penalty can be reduced by spreading all the 8 workers across 4 compute nodes (2 workers per compute node). It is therefore useful to note that in a realistic heterogeneous Hadoop cluster, one is likely to see variable performance based on the hardware of the node used. Performance, as in our scenario, is also impacted by the type of Java virtual machine being used to execute the sentiment analysis algorithm.

Focusing specifically on the deployment strategy, it can be seen that results from the UCLM Cloud have shown a performance increase as the level of distribution of the VMs increases. This difference shows that when they are deployed within the same physical machine (even without CPU and main memory overcommitment), workers interfere with the operation of other workers, resulting in reduced performance. This fact means that running this application on a virtual cluster in a production Cloud environment may suffer perturbations from other VMs deployed within the same physical machine.

Finally, when investigating the variability of the Cloud system for the UCLM Cloud infrastructure observed during the execution of the application (multiple times over 3 days) with 15 million tweets archive (of more than 2 Gb) and different deployment strategies (Table II), it can be observed that there is considerable variability (in terms of execution time) when various VMs are allocated to the same physical machine. This variability is reduced as the VMs are scattered over multiple idle machines within the cluster. Taking 1 worker VM per node as the reference point (it is the best scenario), deploying 2 worker VMs per node reduces the number of physical machines by half, but as a consequence the variability is increased by 1.64 times; allocating 4 worker VMs per node reduces the number of physical machines by four but increases this variability by more than twice, while allocate 8 worker VMs within the same physical machine results in a very high variability. As a consequence, the best compromise between the deployment strategy and amount of physical machines obtained is 4 workers per node. These results have been observed when running the KVM hypervisor on OpenNebula Cloud.

Table II
UCLM CLOUD TWEET PROCESSING APPLICATION VARIABILITY.

# Workers	Execution time variability (Seconds)	Ratio
8 Workers per Node	92.92	12.54
4 Workers per Node	16.13	2.17
2 Workers per Node	12.19	1.64
1 Workers per Node	7.41	Reference

VI. CONCLUSIONS AND FUTURE WORK

The COSMOS system is presented along with issues associated with archiving and analysing data for social media analysis. The novelty of this work stems from two perspectives: (i) the use of a distributed processing environment using Hadoop, executed on an OpenNebula-based Cloud – with the particular intention of understanding how VM location and number impact the overall analysis performance; (ii) for COSMOS archived data, a test application focused on sentiment analysis (that can be re-purposed and adapted with ease) and its performance evaluation. Furthermore, the behaviour of multiple virtual clusters and their deployment strategy is studied on two different types of private Cloud environments.

The main conclusion from this article is the feasibility of using Hadoop and Map/Reduce paradigm for COSMOS archived data distributed computing. We demonstrate the performance benefits achieved by using multiple "virtual" worker nodes, compared to running a sequential version of the application. Although various virtual cluster configurations have different associated performance profiles, the greater the number of workers deployed, the better the performance achieved – limited by the amount of VMs that can run simultaneously without affecting each other. Hence, there is a limit on the number of VMs that can be hosted per physical machine to ensure consistent and repeatable performance. This particular application has primarily focused on the use of text data – using other types of content, such as images, will have a different performance profile. However, it is useful to note that there has been a significant increase in the generation of user generated text data in recent years (twitter, facebook status updates, blogs, etc), and this approach can also apply to any of these types of data sources.

As a consequence from the application of a distributed model (such as Hadoop, which requires a virtual cluster) over a Cloud environment, the deployment policy affects the entire performance, where we have observed perturbations between VMs deployed within the same physical machine, even when no CPU overcommitment is allowed. Furthermore, the variability of Cloud environment for the given application shows that the deployment policy is essential to ensure that predictable performance can be achieved – an aspect that cannot be controlled when deploying over public Clouds.

Future work involves extensing the types of analysis we can carry out, evaluating our approach on a bigger Cloud environment by increasing the size of the virtual clusters and increasing the number of Twitter messages we can analysis. To deploy this system for research users, it would also be necessary to assess the performance of the user for dealing with multiple concurrent users using the same Cloud service, tackling the problem of interference between VMs to reduce performance variability. The latter could be achieved by providing a limit on the maximum number of VMs we should host on a given physical server.

A. *Impact on Social Media Analysis*

The impact of this Hadoop deployment on social media analysis is very significant. It shows we can scale content analysis of Tweets to the extent where we can calculate approximately 5 days worth of Tweets in around 3 minutes, as opposed to $>$ 10 minutes on a single machine. As the COSMOS compute infrastructure grows we expect to be able add more nodes and workers to the problems and bring the processing time down further. Even though the wait-time it is still not as short as we would like, this scaled approach provides academics with computational analytical tools that have thus far only been available through proprietary (and expensive) tools, which are not open to methodological inspection.

ACKNOWLEDGMENT

Part of the "COSMOS" development was carried out through the UK Joint Information Systems Committee (JISC) funded "Supporting Empirical Social Scientific Research with a Virtual Research Environment" project under the Digital Infrastructure: Research Tools Programme. We would like to thank JISC for their support. This work has also included support from the Spanish Government under Grants TIN2009-14475-C04-03 and TIN2012-38341-C04-04 and through a FPI scholarship associated with the TIN2009-14475-C04-03 project.

REFERENCES

[1] P. Anderson, "What is Web 2.0? Ideas, technologies and implications for education," in *JISC Online Report. Available*

at www.jisc.ac.uk/media/documents/techwatch/tsw0701b.pdf, 2007.

[2] A. Iosup, N. Yigitbasi, and D. H. J. Epema, "On the performance variability of production cloud services," in *Proceedings of CCGRID*, 2011, pp. 104–113.

[3] L. Gillam, B. Li, and J. O'Loughlin, "Adding cloud performance to service level agreements," in *Proceedings of CLOSER*, 2012, pp. 621–630.

[4] "SentiStrength: The sentiment strength detection in short texts," Web page at http://sentistrength.wlv.ac.uk/, Last access: 16th July, 2012.

[5] B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," in *Foundations and Trends in Information Retrieval 2(1-2) – available at: http://www.cs.cornell.edu/home/llee/opinion-mining-sentiment-analysis-survey.html*, 2008, pp. 1–135.

[6] "OpenNebula: The Open Source Solution for Data Center Virtualization," Web page at http://opennebula.org/, Last access: 16th August, 2012.

[7] N. Kourtellis, J. Finnis, P. Anderson, J. Blackburn, C. Borcea, and A. Iamnitchi, "Prometheus: user-controlled p2p social data management for socially-aware applications," in *Proceedings of the ACM/IFIP/USENIX 11th International Conference on Middleware*, ser. Middleware '10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 212–231. [Online]. Available: http://dl.acm.org/citation.cfm?id=2023718.2023733

[8] J. Ratkiewicz, M. Conover, M. Meiss, B. Gonalves, S. Patil, A. Flammini, and F. Menczer, "Truthy: mapping the spread of astroturf in microblog streams," in *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011 (Companion Volume)*, S. Srinivasan, K. Ramamritham, A. Kumar, M. P. Ravindra, E. Bertino, and R. Kumar, Eds. ACM, 2011, pp. 249–252.

[9] G. Xiang, B. Fan, L. Wang, J. Hong, and C. Rose, "Detecting Offensive Tweets via Topical Feature Discovery over a Large Scale Twitter Corpus," in *Proceedings of CIKM, October 29 – November 2*. ACM, 2012.

[10] J. Lin and A. Kolcz, "Large-scale machine learning at twitter," in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '12. New York, NY, USA: ACM, 2012, pp. 793–804. [Online]. Available: http://doi.acm.org/10.1145/2213836.2213958

[11] C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins, "Pig latin: a not-so-foreign language for data processing," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, ser. SIGMOD '08. New York, NY, USA: ACM, 2008, pp. 1099–1110. [Online]. Available: http://doi.acm.org/10.1145/1376616.1376726

[12] T. White, *Hadoop: The Definitive Guide*, M. Loukides, Ed. O'Reilly, June 2009.

[13] C. Lam, *Hadoop in Action*. Manning Publications, December 2010.

[14] "BashReduce," Web page at https://github.com/erikfrey/bashreduce, Last access: 12th October, 2012.

[15] "Storm. Distributed and Fault-tolerant Realtime Computation," Web page at http://storm-project.net/, Last access: 12th October, 2012.

[16] P. Burnap, O. Rana, and N. Avis, "Making Sense of Self Reported Socially Significant Data Using Computational Methods," *International Journal of Social Research Methods (to appear)*, 2013.

[17] A. Bruns and Y. Liang, "Tools and methods for capturing Twitter data during natural disasters," *First Monday, April 2, 2012*, vol. 17, no. 4.

[18] "140Dev PHP Library," Web page at http://140dev.com, Last access: 9th October, 2012.

[19] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, "Bigtable: A Distributed Storage System for Structured Data," in *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation - Volume 7*, ser. OSDI '06. Berkeley, CA, USA: USENIX Association, November 2006, pp. 15–15. [Online]. Available: http://dl.acm.org/citation.cfm?id=1267308.1267323

[20] "Alchemy Sentiment Analysis API," Web page at http://www.alchemyapi.com/api/sentiment/, Last access: 11th October, 2012.

[21] M. Thelwall, K. Buckley, G. Paltogou, D. Cai, and A. Kappas, "Sentiment strength detection in short informal text," in *Journal of the American Society for Information Science and Technology*, vol. 61, 2010, pp. 2544–2558.

[22] "Apache Hadoop," Web page at http://hadoop.apache.org/, Last access: 17th August, 2012.

[23] Cloud-B-Lab, "Setting up a multinode hadoop cluster using ubuntu 11.04 32 bit server virtual machines," march 2012.

[24] "Kernel Based Virtual Machine (KVM)," Web page at http://www.linux-kvm.org/, Last access: 15th August, 2012.

[25] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," *SIGOPS Oper. Syst. Rev.*, vol. 37, no. 5, pp. 164–177, Oct. 2003. [Online]. Available: http://doi.acm.org/10.1145/1165389.945462

[26] "VMware Virtualization Software for Desktops, Servers and Virtual Machines for Public and Private Cloud Solutions," Web page at http://www.vmware.com/, Last access: 3rd October, 2012.

[27] "CentOS: The Community ENTerprise Operating System," Web page at http://www.centos.org/, Last access: 10th July, 2012.