

Towards the Cloudification of the Social Networks Analytics

Daniel Cea, Jordi Nin, Rubén Tous, Jordi Torres, and Eduard Ayguadé

Barcelona Supercomputing Center (BSC)
Universitat Politècnica de Catalunya (BarcelonaTech)
Barcelona, Catalonia, Spain
`dcea,nin,rtous,torres,eduard@ac.upc.edu`

Abstract. In the last years, with the increase of the available data from social networks and the rise of big data technologies, social data has emerged as one of the most profitable market for companies to increase their benefits. Besides, social computation scientists see such data as a vast ocean of information to study modern human societies. Nowadays, enterprises and researchers are developing their own mining tools in house, or they are outsourcing their social media mining needs to specialised companies with its consequent economical cost. In this paper, we present the first cloud computing service to facilitate the deployment of social media analytics applications to allow data practitioners to use social mining tools as a service. The main advantage of this service is the possibility to run different queries at the same time and combine their results in real time. Additionally, we also introduce *twearch*, a prototype to develop twitter mining algorithms as services in the cloud.

Keywords: Social Mining, Green Computing, Cloud Computing, Big Data Analytics, Twitter Mining, Stream Processing

1 Introduction

A growing proportion of human activities, such as social interactions, job relationships, entertainment, collaborative working, shopping, and in general, gathering information, are now mediated by social networks and web services. Such digitally mediated human behaviours can easily be recorded and analysed, fuelling the emergence of (1) computational social science, (2) new services such as tuned search engines or social recommender systems, and (3) targeted online marketing. Due to this, public and private sector actors expect to use big data to aggregate all of this data, extract information (and knowledge) from it, and identify value to citizens, clients and consumers [15, 4, 14].

According to one research study [9] from the University of Maryland's Robert H. Smith School of Business, this growing allows Facebook, twitter and other social media sites to create between 182,000 and 235,000 jobs in US and has added between \$12.19 billion and \$15.71 billion in wages and salaries. A similar study funded by Facebook finds that in Europe, Facebook added a similar number of jobs (approximately 232,000). All these business opportunities have been

only possible thanks to the possibility to mine social media insights through development APIs.

While many success stories proliferate, in the private sector, social media analytics have found a *killer application* on the Market Research arena. Market research analyses information about customers and target markets to study the market size, market need and competition. The analysis of social media data provides an unprecedented opportunity to understand how customers behave and why, becoming a key component of business strategy. Platforms for social media analytics are proliferating rapidly nowadays (Twitonomy [20], SumAll [17], TwitSprout [21], etc.), with a recent trend towards specialising on market research and brand strength analysis (Brandchats [1], Brandwatch [2], etc.). However, most of these platforms are private initiatives and the ones that are freely available present important hardware restrictions and, therefore, limitations to perform complex queries.

In order to overcome these aforementioned limitations, the contributions of this paper are: an elastic cloud computing service to facilitate the deployment of social media analytics applications together a graphic framework to automatically display some mining results. The proposed Platform as a Service (PaaS) provides the bottom subsystems of the solution stack required by companies (underlying source API access, storage and retrieval) and provisions the necessary hosting capabilities in a scalable and elastic manner without duplicating computer resources. With our architecture, one client can query different social networks with different queries at the same time and display, in real time, the aggregated mining results.

The rest of this paper is organised as follows. Firstly, in Section 2 a brief overview of the related work is introduced. Secondly, a complete description of the proposed architecture and software stack is depicted in Section 3. Later, in Section 4 a real example for the twitter social network is shown. Finally, the paper finishes with some conclusions and future work.

2 Related Work

The major part of social mining platforms covers the entire lifecycle of data analysis, from data gathering to reporting and visualisation. In order to do so, they spend a lot of effort on *reinventing-the-wheel* at the initial stages (data gathering, storage and querying) shortening their resources for the analysis and visualisation stages [11], in which reside their competitive advantage.

For instance in [22], authors propose one architecture to extract and cluster all the tweets of a city. However, if two cities must be monitored, the architecture must be completely duplicate, posing serious scalability problems. Other platform is Datasift [5], where users pay for executing queries over a large set of data sources, but without any option to execute part of the analysis in-house to save money.

In [8], authors describes SONDY, a tool for analysis of trends and dynamics in online social data using twitter. SONDY is written in java, therefore, it is

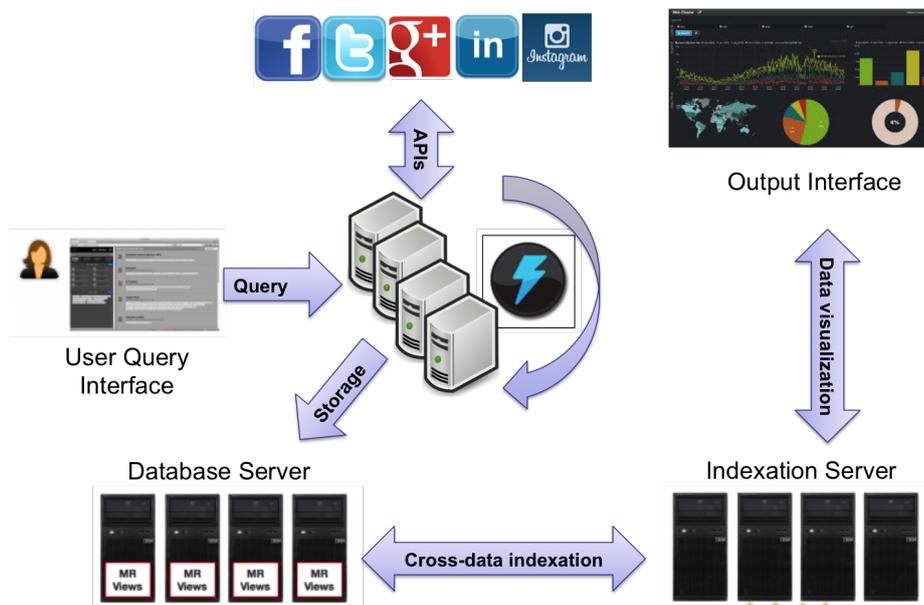


Fig. 1. General Architecture

difficult to make it scale. Besides, it does not allow users to aggregate data coming from several social networks. Finally, SocialSTROM [23] is a cloud-based hub which facilitates the acquisition, storage and analysis of live data from social media feeds (Twitter, Facebook, RSS sources and blogs), as SONDY, it is a java application and it also present scalability limitations.

3 Framework Definition

In this section we detail the main components of our framework, their goals, the selected software and how they interact.

The proposed social mining architecture is composed by 5 independent but interacting elements, as shown in Figure 1, each of them described below. The architecture receives as input the query parameters to be "analysed". Other parameters are optional, such as the possible data post-processing techniques, data enrichment methods, data sources crossings, etc. The architecture outputs some graphical statistics, in parallel data is stored into a NoSQL database for further analysis if needed.

The main components are:

- **User Query Interface:** the user query interface consist on a responsive web application where the user can set the query and also obtain some

- feedback about the execution, such as total number of retrieved elements, query parameters, execution time, server usage, etc. Apart from that, user is also able to recover the queries executed in the past for further analysing.
- **Processing Cluster:** the cluster consists on several distributed nodes that are in charge of retrieving the public information, as well as, to post-process it if needed. For example, in the running example depicted in Section 4, it is responsible to connect to the Twitter Streaming API, manage the persistent HTTP connection, and filter out the results.
 - **Database Server:** the database server is composed of several nodes where data is distributed along different nodes, offering a flexible and scalable data model.
 - **Indexation Server:** indexation server creates a set of data indexes to increase the performance of the database server. It maintains a reverse index for each retrieved word. It automatically updates such indexes when a new data element arrives.
 - **Output Interface:** for the output interface, where the results of the query are displayed, we use the graphic framework Kibana, which offers a responsive and friendly display solution for our analytics.

3.1 Software Stack

In this section we introduce all the software components, from the virtualisation platform to the data visualisation tools that we have used to develop our architecture.

First of all, to easily create and destroy virtual machines we execute Open Nebula [13] in the cluster of the Computer Architecture Department of the Technical University of Catalonia ¹. Open Nebula is an open-source project delivering a simple and flexible solution to build and manage enterprise clouds and virtualized data centers. Combining existing virtualization technologies with features for multi-tenancy, automatic provision and elasticity, open Nebula aims to provide a *open, flexible, extensible, and comprehensive* management layer to automate and orchestrate the operation of enterprise clouds. We have used Open Nebula to deploy the required virtual machines for our architecture. Virtualisation makes our system elastic with regards the amount of data captured in any moment.

For the Database server, we use Couchbase [3] as the distributed data repository. Couchbase is an open-source, distributed, NoSQL document-driven database optimised for interactive applications serving many concurrent users; creating, storing, retrieving, aggregating, manipulating and presenting the data. Couchbase borns from to the fusion of Membase and CouchOne projects in January 2012. The current release offers features including *JSON document store, indexing and querying, incremental MapReduce and cross datacenter replication*.

For the Indexation server, the natural decision is to use Elasticsearch [7], the native indexation software for Couchbase. Elasticsearch is an open-source,

¹ <http://www.ac.upc.edu/serveis-tic/altas-prestaciones>

distributed, real-time search and analytic engine built specifically to run on NoSQL document-driven databases. Documents are stored as JSON, and all the fields are automatically indexed and usable in a single query. Elasticsearch principal features include: scalability, high availability, multi-tenancy, full text search, conflict management between different versions, and a restful API using JSON over HTTP. Elasticsearch permits us to create a large amount of queries over a set of different data streams stored in Couchbase.

Finally, for the output interface, we use Kibana [10] because it is based on javascript and bootstrap and it is fully compatible with any browser. Kibana is an open-source, scalable, real-time visualisation tool natively integrated with Elasticsearch. Its main goal is to display the data stored with Elasticsearch in an elegant graphical manner. Kibana key features include time-based comparisons, easy creation of graphical data representations (plots, charts and maps), flexible, editable and responsive web interface, and a powerful search syntax. In our system, we have adapted our mining methods to display their results in this visualisation framework.

4 Tsearch: A running example

In this section, we describe a proof of concept application to show the feasibility of our architecture. To do that, we have implemented a twitter listener and some basic queries on the top of Kibana. Tsearch offers a simple query interface for twitter able to filter in real time the twitter data stream, by means of any combination of keywords, locations, language, etc. Besides, Tsearch also offers to data miners, an output interface to create graphics using javascript.

4.1 Twitter Connection

Twitter, is an online social network born in March 2006 [6] that enables users to send and read “tweets”, which are text messages limited to 140 characters, also allows data programmers to access in real time to perform any kind of text-mining technique, such as clustering, TF-IDF, etc. However, it is impossible for a single computer to capture and process in real time the complete twitter information flow. For example, in 2012, Twitter had 500 million users registered posting over 350 million Tweets per day [12, 16, 18, 19].

Twitter offers two different APIs for developers:

- **REST API:** used to retrieve past *tweets* based in different filters. There are different resources depending on the data to be retrieved: Accounts, friendships, geolocations, statuses, users, etc. Depending on the resource, the number of queries per account is limited from 15 to 450 per *rate limit window* (by March 2014, 15 minutes long). Each query response contains 100 *tweets*.
- **Streaming API:** used to retrieve *tweets* in real time. Not all the *tweets* are sent, but only a 5%, so it’s mostly used for analysis purposes. There are 3

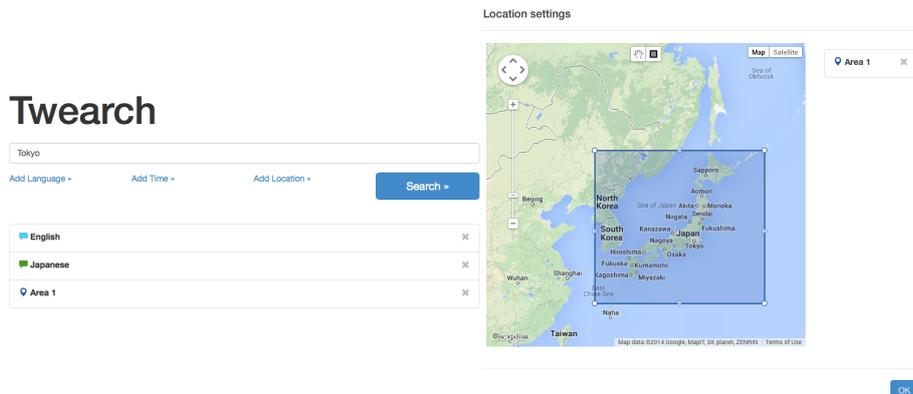


Fig. 2. Input interface query

different resources, depending on the target: Public streams, User streams, and Site streams. The stream API is limited to 1 data stream per account.

Since our framework is designed to manage stream data, we connected Twearch to the Twitter streaming API.

4.2 Query System

As a running example for this paper, we want to retrieve all tweets containing the word “Tokyo” during a week. The input interface, as it is shown in Figure 2, asks Twitter to filter all incoming tweets and sends us only those containing the string “Tokyo”. Apart from keywords, using the twearch interface, one user can filter the twitter stream using hashtags, mentions, languages and spatial coordinates.

Note that, similar input interfaces can be created for others social networks, such as foursquares, instagram o facebook. Doing this, it is easy to cross information coming from different social networks to enrich data analytics without too much effort and allowing the data miner to recycle the listeners for future analytics.

The input interface sends the query to one node of the processing cluster. Once the query is received in one of the nodes, it opens a permanent HTTP connection with the Twitter Streaming API, and the data stream will begin to flow back containing the requested tweets. Then, data stream will be processed, enriched, stored into the Couchbase database and indexed by Elasticsearch server.

When the output interface (Kibana) is opened, each panel will send a predefined query through the Elasticsearch API REST, using HTTP GET requests, to ask for the concrete fields and filters needed for that panel, and then populate the data graphically using different types of panel. Here, it is important to

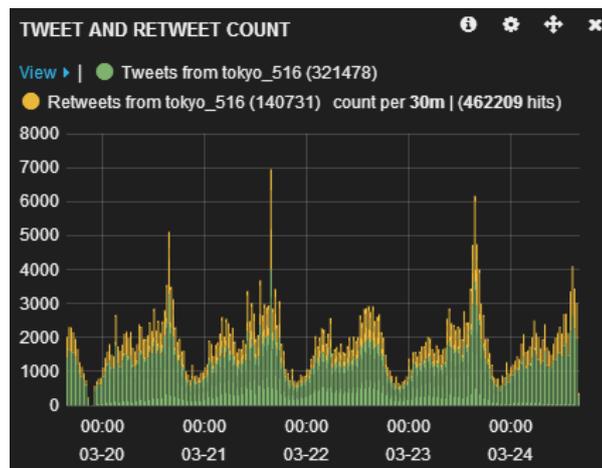


Fig. 3. Number of tweets and retweets per 30 minutes

highlight that more than one query (even from more than one social network) can be sent to Kibana, therefore query results can be aggregated and recycled each time a new query is executed.

As an example, we have created the Tweets/Retweets panel (Figure 3), where there are 2 different queries inside the request: A count of all tweets from the stream, and another count from all tweets from the stream whose 'retweeted' field is true.

We have also implemented other two panels (Figures 4.(a) and (b)) both related to the Tweet text: one counting the top 10 hashtags (a), and another for user mentions (b). This information is obtained after processing the text field of each tweet stored, and separating the strings depending whether they are hashtags or user mentions. From these two panels we can easily extract some knowledge, such as the high social impact of two teenagers groups; *egirls* and *nuest* from Japan and Korea respectively.

Finally, in order to exploit a different type of the information stored in a tweet, we have also the geo-positioned in a world map all the geo-located tweets. To do that, we use the Open Street Maps library, as it is shown in Figure 5.

4.3 Query Refining

Additionally, it is possible to refine the results and focus them on a concrete time period using the Kibana output interface. Data miners can filter the documents by their timestamp and study in detail concrete time periods. For instance, Figure 3 depicts some tweets and re-tweets peaks from 22:00 to 23:00 during some days of March. If we focus on these tweets, as it is shown in Figure 6.(a), it is possible to observe that the mentioned users changes (see Figure 6.(b)).

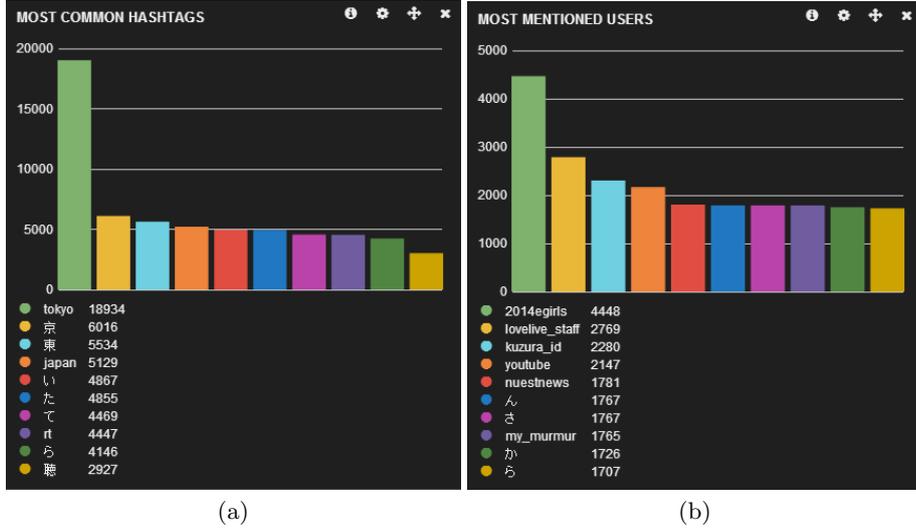


Fig. 4. Counting query of the most common words, hashtags and mentioned users among tweets

Concretely, the most mentioned user is *lovelive* instead of *2014egirls*. *Loveline* is a very popular Japanese anime serial, and the first chapters of its second season were broadcasted during these concretes time periods. So, we were able to detect new trends easily using *twearch*.

Finally, observing Figure 5, it is possible to see that most of the tweets are located in Tokyo, zooming at the center of the city, we discover that a big cluster of tweets is located at the Shibuya station, one of the most visited parts of the city. Therefore, using this map it is possible to automatically detect huge twitter users concentrations for a given query.

4.4 Platform Performance

Apart from the social information extracted from Twitter, another topic of interest is to analyse the performance of the system taking measurements of the main hardware components during the information retrieval process.

To achieve that, the components of the architecture send reports, every 5 seconds, about the amount of CPU and Memory being consumed, Specifically:

- **Processing node:** It is the node of the processing cluster responsible to host the input query interface and display the mining results. Besides, it is also responsible to send the streaming job start signal to the proper crawling node.
- **Crawling node:** Node responsible to connect to Twitter, receive the streaming data, refine the query and store the stream in the database.



Fig. 5. Geopositioned Tweets in OpenStreetMaps

- **CouchBase + ElasticSearch:** Node hosting the Couchbase database and the ElasticSearch indexer.

Those results are stored in CouchBase and displayed in a Kibana histogram, which shows results of the average consumption every 10 minutes during the part of the streaming process (in our case, the last 2 days).

As we illustrate in Figure 8, on the one hand, the amount of CPU used by the Processing and Crawling nodes is almost negligible, with a mean value less than 2% of the CPU with some punctual peaks that are always less than 5%. On the other hand, the Couchbase and ElasticSearch are CPU-consuming processes that average a 15% of CPU consumption with peaks over the 25%

About the memory, as we can observe in Figure 9, all processes keep a constant amount of consumed memory: Couchbase and Elasticsearch use around 550MB of RAM memory, the crawling node is around 100MB, and finally the processing node around 90MB.

The CPU performance results show that it is possible to consolidate several queries into a single virtual machine, reducing the required number of virtual machines needed to perform complex queries where lot of information has to be retrieved. For the data management system (database and indexes) the real bottleneck is the RAM memory (see Figure 9) not the CPU. To overcome this drawback, more than one virtual machines (or physical servers) can be deployed to exploit the scalability of Couchbase and Elasticsearch.

5 Conclusions

In this paper we have described all the components of an elastic and scalable framework for social mining in a cloud infrastructure, in our case Open Nebula. We have described the database management system, the query language and the visualisation tool. Finally as a proof of concept, we have described how to collect

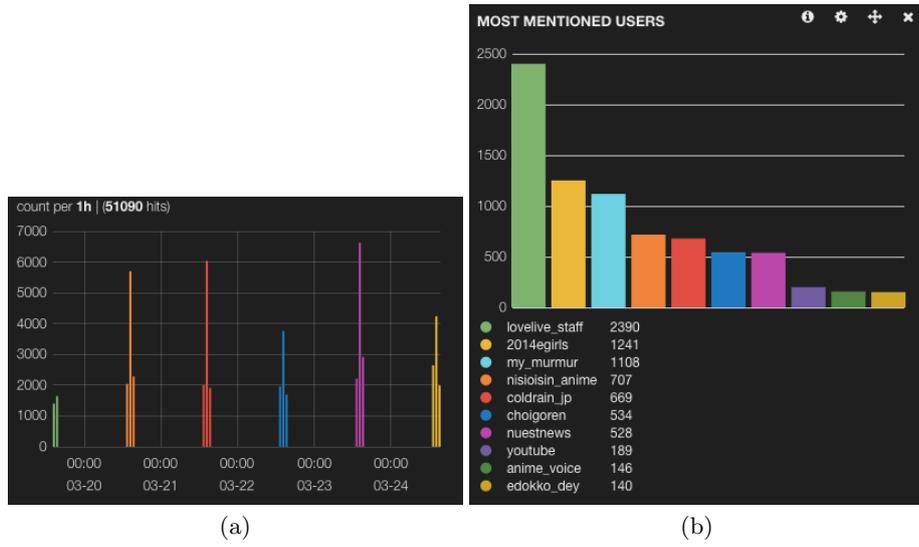


Fig. 6. Lovelive new season advertising.

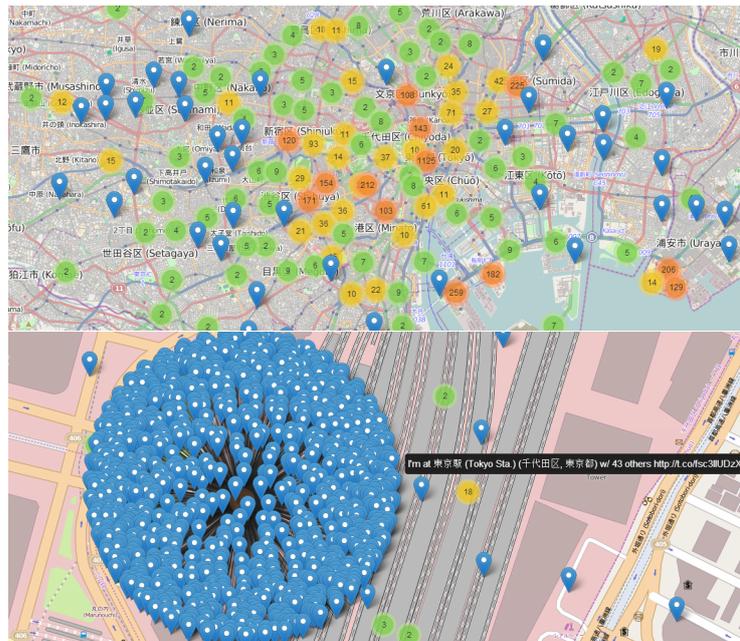


Fig. 7. Shibuya station tweet cluster.

data from twitter and we have displayed some data analytics and performance metrics for a given query using the proposed query system.

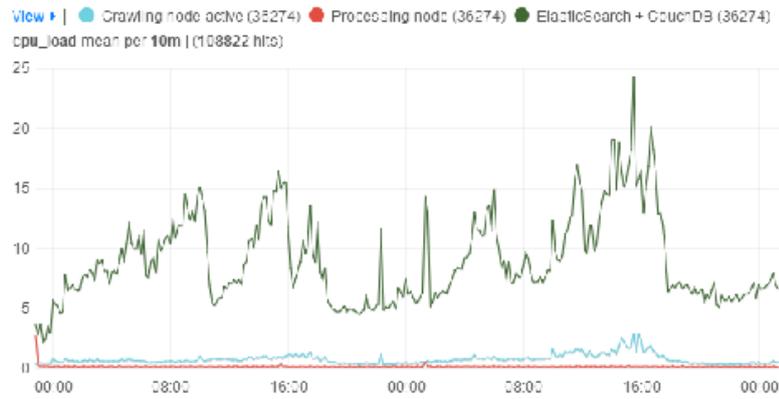


Fig. 8. CPU consumption, average and maximum, of all the system components during a 2 day streaming.

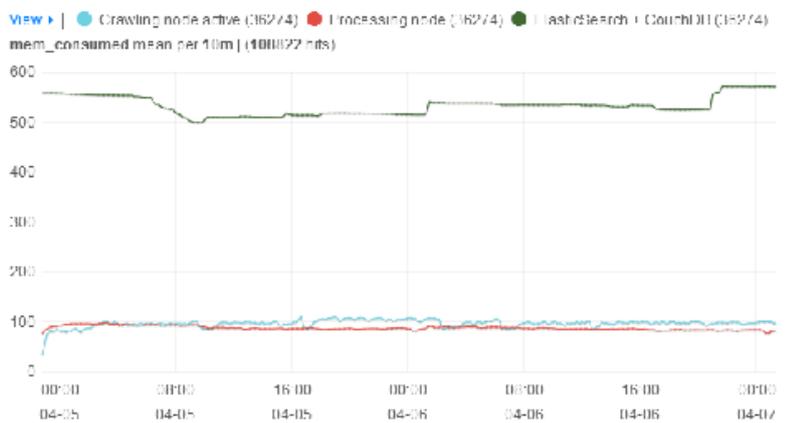


Fig. 9. Memory consumption, average and maximum, of all the system components during a 2 day streaming.

In the near future, we plan to add more functionalities to our platform as for instance, natural language processing methods, automatic data enrichment by means of data crossing. Finally, we would like to define a decision support system to help designers of appliances to optimise resource allocation in a semi-supervised way.

Acknowledgments. This work is partially supported by the Ministry of Science and Technology of Spain under contract TIN2012-34557, by the BSC-CNS Severo Ochoa program (SEV-2011-00067). Besides, authors would like to thank CA

Technologies for funding this research through a collaboration agreement with Universitat Politècnica de Catalunya

References

1. Brandchats. <http://www.brandchats.com> (Accessed March 20, 2014).
2. Brandwatch. <http://www.brandwatch.com> (Accessed March 20, 2014).
3. Martin C. Brown. *Getting Started with Couchbase Server - Extreme Scalability at Your Fingertips*. O'Reilly, 2012.
4. Junghoon Chae, Dennis Thom, Yun Jang, SungYe Kim, Thomas Ertl, and David S Ebert. Public behavior response analysis in disaster events utilizing visual analytics of microblog data. *Computers & Graphics*, 38:51–60, 2014.
5. Datasift. <http://datasift.com> (Accessed March 20, 2014).
6. J. Dorsey. just setting up my twttr. *Twitter. Oldest Tweet ever from Twitter founder.*, 2006.
7. Elasticsearch official website. <http://www.elasticsearch.org/> (Accessed March 20, 2014).
8. C. Adrien Guille, H. Hacid, and D. Zighed. Soudy: An open source platform for social dynamics mining and analysis. In *ACM SIGMOD*, 2013.
9. Il-Horn Hann, Siva Viswanathan, and Byungwan Koh. *The Facebook App Economy*. Center for Digital Innovation, University of Maryland, 2011.
10. Kibana website. <http://www.elasticsearch.org/overview/kibana/> (Accessed March 20, 2014).
11. Michael Lieberman. Visualizing big data: Social network analysis. In *Digital Research Conference*, 2014.
12. I. Lunden. Twitter passed 500m users in june 2012, 140m of them in us; jakarta biggest tweeting“ city. *techcrunch*, 2012.
13. Opennebula key features and functionality. <http://www.opennebula.org> (Accessed March 20, 2014).
14. Taylor Shelton, Ate Poorthuis, Mark Graham, and Matthew Zook. Mapping the data shadows of hurricane sandy: Uncovering the sociospatial dimensions of 'big data'. *Shelton, T., Poorthuis, A., Graham, M., and Zook, M.*, 2014.
15. Mark A Stoové and Alisa E Pedrana. Making the most of a brave new world: Opportunities and considerations for using twitter as a public health monitoring tool. *Preventive Medicine*, 2014.
16. D. Strachan. Twitter: How to set up your account. *The Daily Telegraph*, 2009.
17. Sumall. <http://sumall.com> (Accessed March 20, 2014).
18. Twitter Search Team. The engineering behind twitter’s new search experience. Technical report, Twitter Engineering Blog, 2011.
19. Twitter Search Team. Twitter turns six. Technical report, Twitter Engineering Blog, 2012.
20. Twitonomy. <http://www.twitonomy.com> (Accessed March 20, 2014).
21. Twitsprout. <http://twitsprout.com> (Accessed March 20, 2014).
22. D. Villatoro, J. Serna, V. Rodríguez, and M. Torrent-Moreno. The tweetbeat of the city: Microblogging used for discovering behavioural patterns during the mwc2012. In *Citizen in Sensor Networks*, volume 7685 of *Lecture Notes on Artificial Intelligence*, pages 43–56. Springer-Verlag, 2012.
23. R. Wood, I. Zheludev, and P. Treleaven. Mining social data with ucl’s socialstorm platform. Technical report, University College of London (UCL), 2011.