2015 7th International Conference on Cyber Conflict:
Architectures in Cyberspace
M.Maybaum, A.-M.Osula, L.Lindström (Eds.)
2015 © NATO CCD COE Publications, Tallinn

Permission to make digital or hard copies of this publication for internal use within NATO and for personal or educational use when for non-profit or non-commercial purposes is granted providing that copies bear this notice and a full citation on the first page. Any other reproduction or transmission requires prior written permission by NATO CCD COE.

Supporting Sense-Making and Decision-Making Through Time Evolution Analysis of Open Sources*

Andrea Balboni
Interdepartment Research Center on Security
University of Modena and Reggio Emilia
Modena, Italy
andrea.balboni@unimore.it

Michele Colajanni
Interdepartment Research Center on Security
University of Modena and Reggio Emilia
Modena, Italy
michele.colajanni@unimore.it

Mirco Marchetti
Interdepartment Research Center on Security
University of Modena and Reggio Emilia
Modena, Italy
mirco.marchetti@unimore.it

Andrea Melegari
Expert System s.p.a.
Modena, Italy
amelegari@expertsystem.com

Abstract: Modern societies produce a huge amount of open source information that is often published on the Web in a natural language form. The impossibility of reading all these documents is paving the way to semantic-based technologies that are able to extract from unstructured documents relevant information for analysts. Most solutions extract uncorrelated pieces of information from individual documents; few of them create links among related documents and, to the best of our knowledge, no technology focuses on the time evolution of relations among entities. We propose a novel approach for managing, querying and visualizing temporal knowledge extracted from unstructured documents that can open the way to novel forms of sense-making and decision-making processes. We leverage state-of-the-art natural language processing engines for the semantic analysis of textual data sources to build a *temporal graph database* that highlights relationships among entities belonging to different documents and time frames. Moreover, we introduce the concept of *temporal graph query* that analysts can use to identify all the relationships of an entity and to visualize their evolution over time. This process enables the application of statistical algorithms that can be oriented to the automatic analysis of anomalies, state change detection, forecasting. Preliminary results demonstrate that the representation of the evolution of entities and relationships allows an analyst to highlight relevant events among the large amount of open source documents.

Keywords: *open sources, semantic analysis, temporal query, sense-making, decision-making*

* This project has been funded with support from the European Commission. Co-funded by the Prevention, Preparedness and Consequence Management of Terrorism and other Security-related Risks Programme of the European Union. This publication reflects the views only of the author, and the European Commission cannot be held responsible for any use which may be made of the information contained therein.

1. INTRODUCTION

Decision-making and sense-making processes take advantage of actionable intelligence gathered from any available information source. The increasing volume of information that analysts can access from the Web augments the importance of *Open-Source Intelligence* (OSINT) (Glassman & Kang, 2012). It is impossible for humans to manage the huge amount of information published on a daily basis as unstructured text documents, possibly written in many different languages. Hence, analysts often rely on software for the semantic analysis of natural language (Baldini, Neri, & Pettoni, 2007) (Neri, Aliprandi, & Camillo, 2011) (Steele, 2007) (Richard A. Best, 2008) (Best, 2008). Modern semantic technologies support OSINT through several features, such as topic detection, categorization and mining of entities and relationships. However, these operations are mostly oriented towards *intra-document relationships* and do not take into account *inter-document relationships* and their dynamics in time.

In this paper, we propose a novel scalable architecture for processing the output produced by semantic engines for natural text analysis and that guarantees the following novel features:

- Detection of relationships among entities that occur in the same document *(intra-document relationships)*;
- Detection of relationships among entities that occur in different documents *(inter-document relationships)*;
- Analysis of how entities and relationships evolve over time;
- Extraction of quantitative numerical data that can be analyzed through statistical algorithms.

We have designed and implemented a prototype that fully implements the processing architecture proposed in this paper and that can execute expressive queries over huge volumes of documents to easily extract information related to entities and their relationships and describe how they evolve over time. This work poses the basis for novel forms of sense-making and decision-making supported by algorithms for graph analysis and by statistical algorithms for the analysis of time series, that can be tailored to anomaly detection, state change detection and forecasting.

The remainder of the paper is organized as follows. Section 2 illustrates the main components of the architecture and the foundations of the semantics analysis technologies employed. Sections 3 and 4 describe the most important components of the proposed architecture: the parser that processes the output of the semantic engine, the *temporal graph database* that organizes intra- and inter-document relationships, the parallel query operations executed among the relevant graph databases. Section 5 presents experimental results obtained through a prototype. Section 6 discusses related work. Section 7 contains concluding remarks and some directions for future work.

2. ARCHITECTURE DESIGN

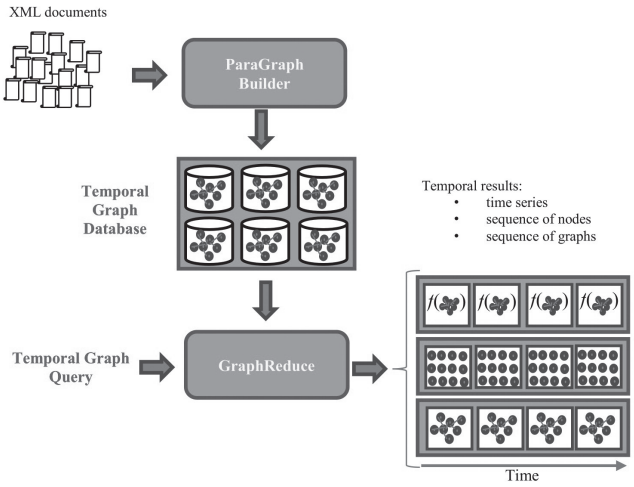
The semantic analysis of natural language is a fundamental enabler for any text analytics methodology. However, traditional engines for semantic analysis suffer some drawbacks that become more evident as the number of documents to analyze increases. For example, after processing large corpora of documents, a semantic engine produces a huge volume of annotated documents (usually in XML format) that needs further processing for scalable storage, indexing and querying. Moreover, since annotated documents are not mutually linked, it is difficult to identify all the facts that involve a given entity and that are described in several different documents.

These motivations induced us to design a novel processing architecture for supporting analysts in storing, connecting and querying all the information produced by engines for semantic analysis. The proposed architecture is based on three main design principles enlisted below:

- The system must be inherently scalable, and designed to run on modern hardware leveraging multicore architectures, distributed file systems and parallel processing of huge amounts of data.
- The second design principle is the focus on relationships among entities across large document sets, rather than on a single document. Indeed, one of our main goals is to establish inter-document relationships, thus allowing analysts to identify all the facts/events involving the same entities or linked by the same relationships. To this end, the information contained in the semantically annotated text documents (produced by semantic engines) are modelled as graphs that contain entities, their relationships and other relevant elements providing information on the context. This design choice has a twofold advantage: it enables the creation of inter-document relationships by connecting graphs related to different documents; it allows us to leverage the state-of-the-art on graph analytics, management and visualization algorithms (Nisar, Fard, & Miller, 2013) (Nguyen, Lenharth, & Pingali, 2013).
- The third design principle is the introduction of the notion of time. With a temporal reference, it is possible to perform novel forms of analysis that show how relationships among entities have evolved, thus giving analysts some insights about the phenomena of interest.

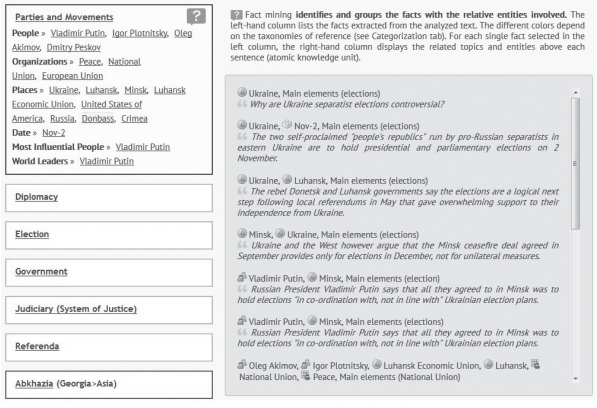
The proposed architecture also supports the extraction of time series that can be analyzed through several statistical algorithms with the aim of eliminating noise (Tosi, Casolari, & Colajanni, 2013), identifying anomalies (Chandola, Banerjee, & Kumar, 2009) and correlations among time series (Esling & Agon, 2012) (Hamilton, 1994) and forecasting their evolution (Brockwell & Davis, 2002). The main components of the processing architecture and its information flow are illustrated in Figure 1.

FIGURE 1: INFORMATION FLOW AND MAIN COMPONENTS OF THE PROPOSED ARCHITECTURE.



Annotated XML documents produced by a semantic engine represent the input of the proposed processing architecture. A semantic engine produces structured information according to different *information channels*. The semantic engine used by our prototype is Cogito Intelligence API (Expert System s.p.a., 2014) which provides 13 information channels that rely on specific taxonomies. The *Fact Mining* category includes specific taxonomies such as *Intelligence, Cyber, Crime* and *Geography*. A *fact* is a collection of entities contained in parts of text (e.g. sentences) categorized according to a set of predefined *domains* and may be described by one or more *topics* (in the *domain*). Focused analyses can be executed by evaluating entities and their relationships in a context involving domains and topics of interest. Figure 2 shows the output of the Fact Mining engine on open source content (http://www.bbc.com/news/world-europe-29831028) obtained with the free web demo of Cogito Intelligence API (Expert System s.p.a., 2015).

FIGURE 2: DEMO WEB APPLICATION OF THE COGITO INTELLIGENCE API.



The *ParaGraph Builder* component analyzes each annotated document, extracts the intra-document graph and populates the *Temporal Graph Database* by merging intra-document graphs into dynamic inter-document graphs. The Temporal Graph Database represents the main information repository that analysts can query to extract useful information. Operations of the ParaGraph builder are described in Section 3. Knowledge extraction is performed through *Temporal Graph Queries* that analysts can submit to another key component called GraphReduce. *GraphReduce* interprets all the temporal queries and processes dynamic inter-document graphs that are stored in the Temporal Graph Database to produce temporal results. Depending on the query submitted by the analysts, temporal results can take three main forms:

- If a function is used to extract a numerical value from graphs, the result is a time series in which each element represents the numerical value computed on the resulting graph in the corresponding timeframe;
- If the temporal query considers entities and not relationships (e.g., by asking for all the entities that are related to Al-Qaeda), then the temporal results are a sequence of sets of entities, that are represented by nodes of a graph;
- If the temporal query aims to represent how entities and their relationships evolve over time, then the temporal results are a sequence of graphs.

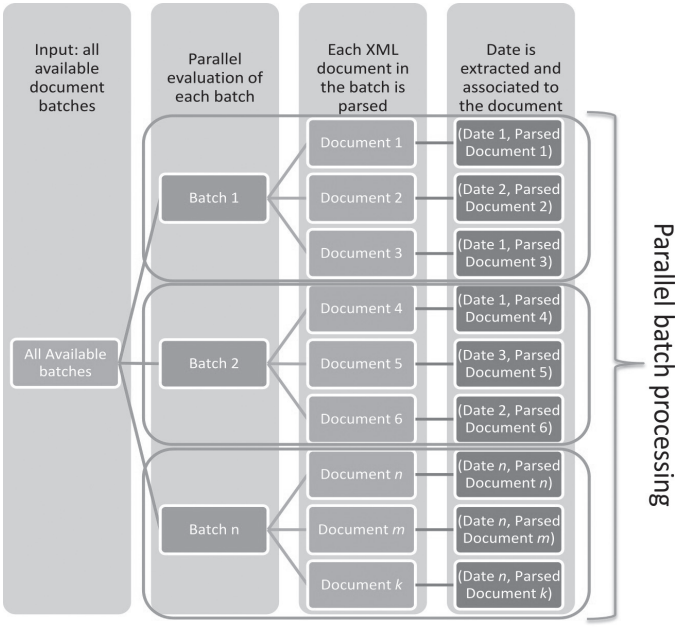
Operations of GraphReduce are described in Section 4.

3. PARAGRAPH BUILDER

The ParaGraph builder analyzes the annotated XML documents produced by the semantic engine and populates the Temporal Graph Database in a two-stage processing pipeline:

- parallel document parsing;
- graph database population.

FIGURE 3: PARALLEL DOCUMENT PARSING PIPELINE OF THE PARA GRAPH BUILDER



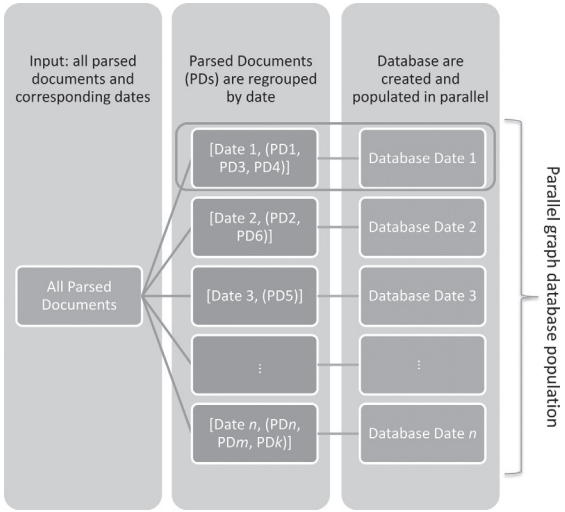
The parallel document parsing pipeline is depicted in Figure 3. In this phase annotated document batches are processed in parallel by the ParaGraph builder, thus guaranteeing high scalability and efficient use of resources in multicore computational architectures.

For each parsed document (PD) in the input batch, the ParaGraph Builder outputs the intra-document graph containing a temporal reference and all the entities and relationships that belong to the Intelligence, Crime and Geography information channels. Any analysis of the evolution of facts, entities and relationships over time relies on the ability to determine the timeframe. This activity can be performed reliably for certain classes of documents, such as news published by on-line newspapers and press agencies on the Web or as RSS feeds. Precise dating of other classes of documents is still an open research area. Each parsed document is associated to (a) the source document, (b) *facts* identified by the semantic analyzer, (c) *topics* and *entities* associated to facts. Parsed documents represent the input for the second processing stage represented in Figure 4. All intra-document graphs whose time reference belong to the same timeframe (e.g. the same calendar day) are aggregated to form a group. Each group contains intra-document graphs that the ParaGraph Builder stores within the same graph database. If a Temporal Graph Database for the timeframe corresponding to a group already exists, the ParaGraph Builder incrementally adds all the new inter-document graphs to the existing group. In particular, it fuses intra-document graph with the other graphs that contain the same entities, thus creating an inter-document graph whose entities and relationships are gathered from many different documents. If the semantic engine is able to recognize that the same entity appears in different documents, even if with different names, all the instances of

the same entity will be fused together. On the other hand, if a graph database for a target timeframe does not exist, the ParaGraph builder creates a new graph database and populates it by inserting all the intra-document graphs contained in the group. All graph databases are populated in parallel, thus ensuring scalability and efficient resource usage on modern multicore architectures and distributed file systems. The set of all the graph databases that store inter-document graphs related to all timeframes forms the Temporal Graph Database and represents the primary knowledge base of the proposed processing architecture.

The graph database used in or prototype implementation is Neo4j (Holzschuher & Peinl, 2013), characterized by a large user community, high performance and scalability, and the support for a powerful graph query language. The Temporal Graph Database is a collection of instances of Neo4j databases. Each graph database instance can be queried independently.

FIGURE 4: PARALLEL GRAPH DATABASE POPULATION PIPELINE OF THE PARAGRAPH BUILDER.



4. GRAPHREDUCE

GraphReduce provides the interface between analysts and all the data stored in the Temporal Graph Database. To extract information from the Temporal Graph Database, the analyst interacts with GraphReduce by issuing queries that are executed in parallel among all the relevant graph databases. When all the parallel queries terminate, GraphReduce collects the partial results and merges them into a single result that is presented to the analyst.

Graph databases are queried through specific query languages that differ from standard query languages for relational databases. In particular, query languages for graph databases focus on the relationships among entities, and are able to express the concept of direct and indirect

connections among entities. Graph queries on Neo4j are expressed via the Cypher (Holzschuher & Peinl, 2013) language. An example of query expressed in Cypher is given in Figure 5.

FIGURE 5: EXAMPLE OF GRAPH QUERY EXPRESSED IN CYPHER QUERY LANGUAGE.

```
MATCH
({key:"Infrastructures"})
      -[:TYPE_OF]->
(infras:Entity)
      -[infras_in_document:ENTITY_OF]->
(document:Document)
      <-[:DOMAIN_OF]-
(:Domain {key:"Act of Terror"})
MATCH
(document)
      <-[person_in_document:ENTITY_OF]-
(person:Entity)
      <-[:TYPE_OF]-
({key: "People"})
RETURN
infras_in_document, person_in_document
```

This query runs over a graph database and returns a sub-graph that represents all the entities of type “People” and “Infrastructures” that participate in facts belonging to the domain “Act of Terror”. It comprises two sub-queries, each introduced by the “MATCH” keyword. The first sub-query identifies all the documents that satisfy two properties: 1) they are connected through a relationship of type “DOMAIN_OF” to the domain “Act of Terror”; 2) they are connected through relationships of type “ENTITY_OF” to entities whose type is “Infrastructures”. We can see from Figure 5 that relationships are represented by arrows and are described by couples “NAME:TYPE” within square brackets (-[NAME:TYPE]->). The name is not mandatory, but can be used to reference the same relationship or the same entity in other parts of the query. As an example, all documents that satisfy the two constraints expressed in the first sub-query are given the name “document”, while all relationships that connect an entity of type “Infrastructure” to a document are referenced by the name “infras_in_document”. The second sub-query expresses a further constraint by selecting only documents that contain entities of type “People”. The relationships returned in the resulting sub-graph are named “person_in_document”.

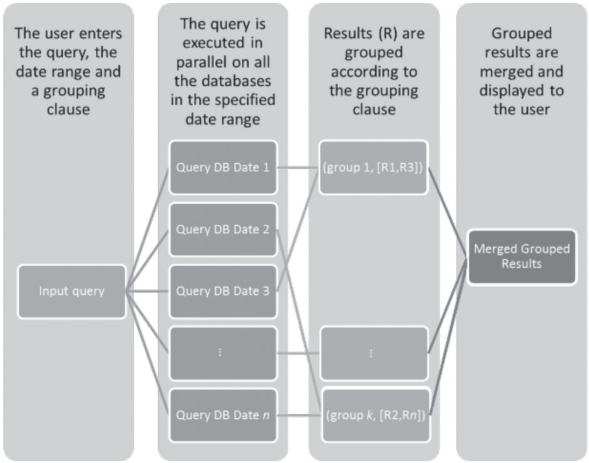
After selecting a subset of entities and relationships through “MATCH” keywords, Neo4j produces an output that contains all the relationships named “infras_in_document” and “person_in_document”, as prescribed by the last part of the query introduced by the keyword “RETURN”. Since a relationship implicitly includes the connected entities, the result produced by Neo4j is a graph.

While Cypher is a powerful query language, it lacks specific keywords to express temporal constraints. Hence, we defined a new class of queries, called *Temporal Graph Queries*, that enrich Cypher by introducing four new keywords that express temporal constraints. Temporal Graph Queries are represented via standard JSON (JavaScript Object Notation) data structures (Ihrig, 2013). A temporal query includes four different elements:

- The first element, introduced by the keyword “query”, represents a Cypher graph query. In principle, it could be expressed in any other language used to query graphs, such as the Gremlin query language (Tausch, Philippsen, & Adersberger, 2011);
- The second and third elements are the *start date* and the *end date* that correspond to the keywords “sdate” and “edate”, respectively. These two keywords define the timeframe over which the query has to be executed;
- The fourth element is the *grouping clause*, expressed by the “groupClause” keyword. For example, a grouping clause “byMonth” means that all the temporal results are grouped on a monthly basis. If a selected timeframe consists of twenty-four months, the temporal results are grouped in twenty-four different graphs, each representing the data related to one calendar month.

A query result is always a series of homogeneous “objects” that can be graphs, sets of entities or numerical values. The particular format depends on the analysis focus.

FIGURE 6: MAIN PROCESSING PHASES OF GRAPHREDUCE.



Graphs are best suited to analyze how the relationships among entities evolve over time; sets of entities are good at identifying which entities (e.g., people, infrastructures, organizations) are mentioned in facts belonging to specific domains; numerical values can be used to build time series that represent the evolution over time of any metric that can be extracted from data.

Temporal queries are received by GraphReduce that executes them efficiently over all the graph databases that belong to the Temporal Graph Database.

Figure 6 shows the main processing steps performed by GraphReduce while executing a temporal graph query. Start and end dates included in the temporal query are used to select only the subset of graph databases that is relevant for the query. The Cypher query included in the

temporal query is executed in parallel over all the relevant graph databases. This design choice ensures high scalability and performance.

Since the set of relevant databases depends only on the timeframe identified in the temporal query, the execution time of the query does not depend on the overall size of the Temporal Graph Database. As the result, queries are very efficient because the size of each graph database group is relatively small (an experimental evaluation of the performance and hardware requirements of the prototype is proposed in Section 5).

Temporal queries expressed over large timeframes require the parallel execution of the same query over a high number of graph databases. However, each graph database is queried independently, and the time required to execute the query over all the graph database is limited by the number of queries that can be executed concurrently. Since the proposed architecture is highly parallelized, it is possible to improve performance by leveraging modern distributed file systems and an appropriate number of processing nodes.

After completion of all the parallel graph queries, GraphReduce collects all results and groups them according the grouping clause expressed in the temporal query. Grouped results are merged in an ordered sequence of objects that is presented to the analyst.

5. EXPERIMENTAL EVALUATION

The experimental testbed used to evaluate the performance of our prototype consists of three main elements: two processing nodes, both equipped with two Intel Xeon E5_2620 CPUs, each with 6 physical cores and support for hyperthread technology, for a total of 24 logical processors, and 16 GB of RAM; a storage node composed of a Storage Area Network (SAN) Fujitsu Eternus DX80 S2 including 36 SAS hard drives, with a rotational speed of 15k rpm and 600 GBytes of size. The logical volume used to store the Temporal Graph Database contains 8 physical disks in a high performance RAID 1+0 configuration, for a total of 2.1 TB storage space.

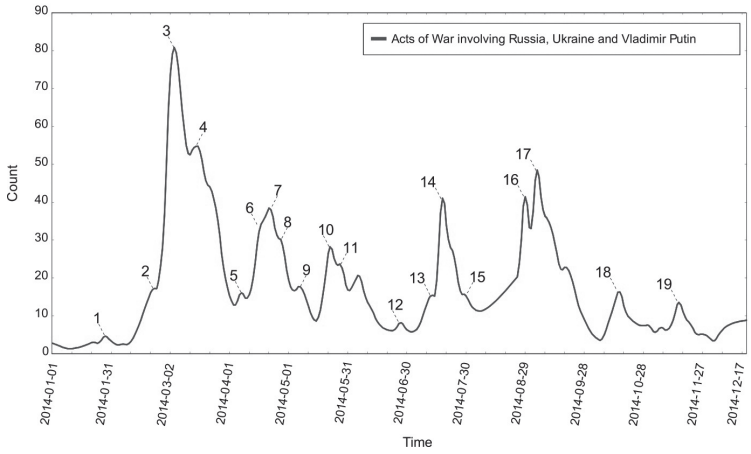
The operating system installed on the processing nodes is Debian Jessie GNU/Linux. The graph databases are implemented through Neo4j version 2.1.2. The ParaGraph builder and GraphReduce components were developed in the Scala programming language (Odersky, Spoon, & Venners, 2008) and leverage the Apache Spark (Zaharia, Chowdhury, Franklin, Shenker, & Stoica, 2010) in-memory map reduce framework (Dean & Ghemawat, 2008).

We tested the prototype by processing 500K annotated documents produced by Cogito. Source documents in natural language are represented by news that can be freely downloaded from the websites and RSS feeds of the major international newspapers and press agencies. These documents were downloaded daily for 18 months. The resulting Temporal Graph Database has a size of 18 GBytes and includes 1835 distinct databases having 676403 distinct entities. Analyzing all the documents and rebuilding the Temporal Graph Database from scratch requires about 60 minutes. All the results presented in this section rely on this dataset.

We present three use cases that demonstrate some of the most important features of the proposed system, and its support for analysis. As a first use case, we consider the ability of our system to create timelines that reflect the popularity of a particular fact in the news. As an example, Figure 7 contains a timeline representing the amount of news that contain facts belonging to the domains “War” and “War Crime” involving Russia, Ukraine and Vladimir Putin. The temporal query considers all the news created in year 2014 with a granularity of three days. To eliminate noise and to provide a better visualization, the system applies an exponential moving average filter (Holt, 2004).

In order to appreciate the capability of the system to produce a significant time series that captures the most important facts related to a given query, we use as our ground truth the list of facts that are included in the timeline of the Russia-Ukraine crises published by the BBC (BBC News Europe, 2015). Selected events from the BBC timeline are associated to numbers (from 1 to 19) which correspond to specific points in the time series. The associations between facts and numbers is given in Table 1. Figure 7 highlights that relevant events are associated to local maximums and abrupt changes in the slope of the timeline. Hence, an analyst can issue a query to pinpoint when relevant facts occurred, and then focus on the analysis of facts that have occurred at specific times.

FIGURE 7: TIME SERIES REPRESENTING THE NUMBER OF NEWS RELATED TO ACTS OF WAR BETWEEN RUSSIA AND UKRAINE INVOLVING VLADIMIR PUTIN OVER THE YEAR 2014.



The preceding query provides an example of a realistic question that could be expressed by an analyst interested in studying the evolution of the crisis between Russia and Ukraine, and of the role of Vladimir Putin. We stress that a similar query cannot be easily expressed through other tools commonly used by analysts. Indeed, an equivalent keyword-based search would be much less effective and would return an endless list of documents ordered by custom ranking algorithms (in a black box approach) that not necessarily reflect the expectations of the analyst. The prototype is able to execute the query and generate the time series in less than 3 minutes on

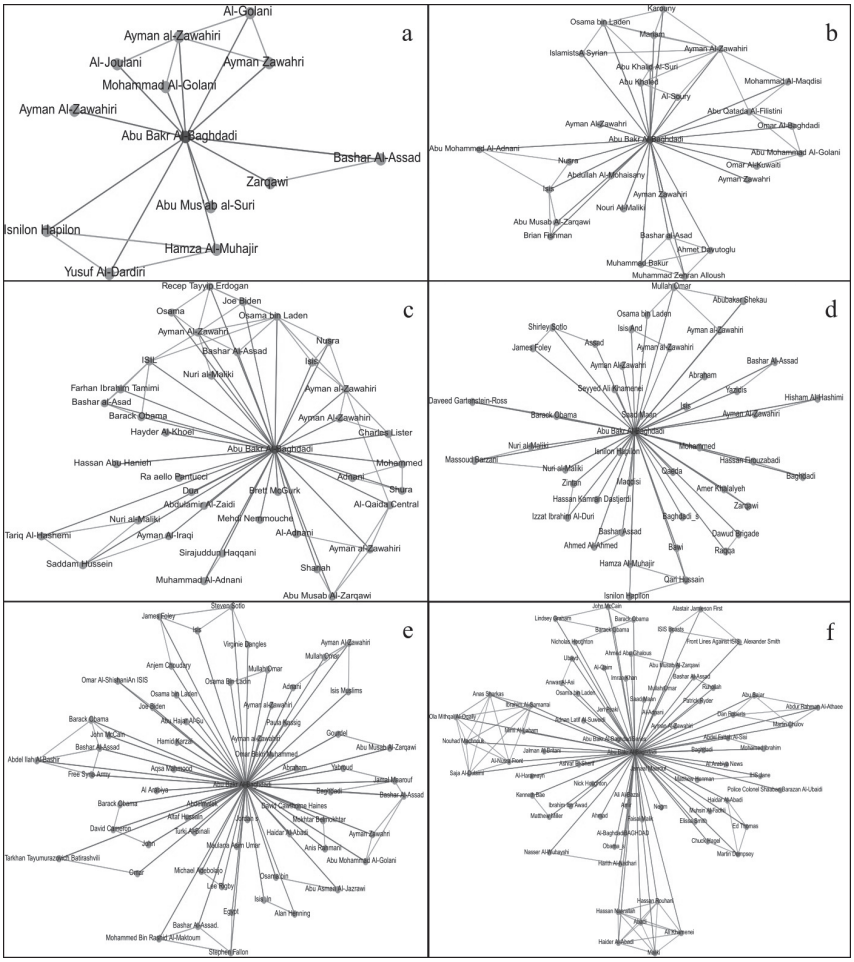
a platform with medium-low computational power. Using a more powerful architecture, we can offer to an analyst the ability to issue queries and visualize the results interactively.

TABLE 1: RELEVANT EVENTS RELATED TO THE RUSSIA-UKRAINE CRISIS PUBLISHED BY THE BBC.

#	Fact
1	Prime Minister Mykola Azarov resigns and parliament annuls the anti-protest law. Parliament passes amnesty bill but opposition rejects conditions.
2	Kiev sees its worst day of violence for almost 70 years. At least 88 people are killed in 48 hours. Video shows uniformed snipers firing at protesters holding makeshift shields.
3	Russia's parliament approves President Vladimir Putin's request to use force in Ukraine to protect Russian interests.
4	President Putin signs a bill to absorb Crimea into the Russian Federation.
5	Protesters occupy government buildings in the east Ukrainian cities of Donetsk, Luhansk and Kharkiv, calling for a referendum on independence. Ukrainian authorities regain control of Kharkiv government buildings the next day.
6	Ukraine's acting President, Olexander Turchynov, announces the start of an "anti-terrorist operation" against pro-Russian separatists. It quickly stalls.
7	Russia, Ukraine, the US and the EU say they have agreed at talks in Geneva on steps to "de-escalate" the crisis in eastern Ukraine. Three people are killed when Ukrainian security forces fend off a raid on a base in Mariupol - the first violent deaths in the east.
8	Ukraine's acting president orders the relaunch of military operations against pro-Russian militants in the east.
9	Clashes in the Black Sea city of Odessa, leave 42 people dead, most of them pro-Russian activists. Most die when they are trapped in a burning building.
10	News coverage about upcoming elections in Ukraine
11	Ukraine elects Petro Poroshenko as president in an election not held in much of the east.
12	Russia's parliament cancels a parliamentary resolution authorising the use of Russian forces in Ukraine.
13	Rebels abandon their command centre at Sloviansk in the face of a government offensive.
14	Malaysia Airlines flight MH17 from Amsterdam is shot down near the village of Grabove in rebel-held territory, with the loss of 298 lives.
15	The EU and US announce new sanctions against Russia.
16	Rebel leader Alexander Zakharchenko says there are 3-4,000 Russian civilians in rebel ranks as the separatists open up a front on the Sea of Azov and capture Novoazovsk.
17	Ukraine and pro-Russian rebels sign a truce in Minsk.
18	President Putin orders thousands of troops stationed near the Ukrainian border to return to their bases.
19	Nato commander Gen Philip Breedlove says Russian military equipment and Russian combat troops have been seen entering Ukraine in columns over several days.

Another interesting example is the study of the social network of a given entity. Figure 8 shows the results of a temporal query that aims at identifying the connections between Abu Bakr Al-Baghdadi and all other persons that are involved in the same facts in the year 2014.

FIGURE 8: SERIES OF GRAPHS REPRESENTING THE EVOLUTION OF THE SOCIAL NETWORK OF ABU BAKR AL-BAGHDADI FOR THE YEAR 2014. EACH GRAPH REPRESENTS TWO MONTHS.



The temporal query groups results on a bimonthly basis, therefore Figure 8 includes six graphs representing the social network of Abu Bakr Al-Baghdadi in different periods of the year (Figure 8.a refers to January and February, 8.b to March and April, and so on). Even without applying complex graph analysis algorithms, it is evident how the social network grows over time. An analyst could infer that the person under analysis has a growing importance, and there are an increasing number of contacts that could be further explored. A quantitative analysis of the numbers of vertexes and edges of each sub-graph that supports this conclusion is represented in Table 2.

TABLE 2: NUMBER OF VERTEXES AND EDGES OF GRAPHS SHOWN IN FIGURE 8.

	Number of direct connections (degree) of Abu Bakr Al-Baghdadi	Total number of connections (edges) of the social network
January-February (a)	12	21
March-April (b)	27	60
May-June (c)	37	79
July-August (d)	43	64
September-October (e)	60	95
November-December (f)	72	128

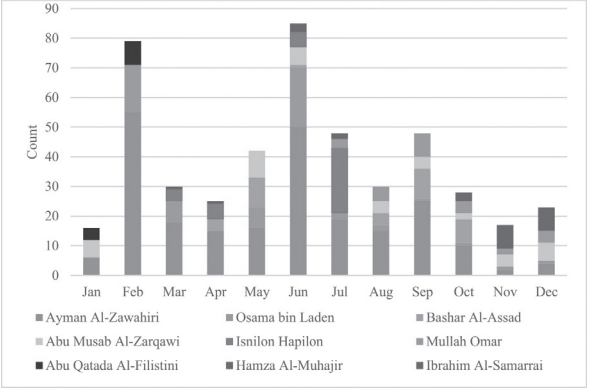
Graphs also highlight different types of relationships. Blue edges identify direct contacts between Abu Bakr Al-Baghdadi and other persons that participate to the same facts, while edges in grey represent direct contacts among persons that are related to Abu Bakr Al-Baghdadi. This representation helps analysts in identifying groups of persons that are directly related to the target node (Abu Bakr Al-Baghdadi) and also triadic closures.

Finally, we present a different graphical representation that can aid analysts in highlighting the relationships between a person and her most active contacts. Figure 9 shows a stacked histogram that represents the number of facts in which Abu Bakr Al-Baghdadi was mentioned together with his top 9 contacts.

From this representation, it is possible to differentiate easily between:

- stable contacts, that are present in all months (such as Ayman Al-Zawahiri);
- recurrent contacts, that are not always present but that appear throughout the year (such Abu Musab Al-Zarqawi);
- old contacts, that were stable or recurrent but stop appearing at a certain point in time (such as Abu Qata Al-Filistini);
- new contacts, that were not present in the past and start appearing at some point in time (such as Ibrahim Al-Samarrai).

FIGURE 9: EVOLUTION OF THE NUMBER OF CO-OCCURRENCES BETWEEN ABU BAKR AL-BAGHDADI AND HIS TOP 9 CONTACTS.



Moreover, by considering the height of the segment associated to a person, it is possible to highlight trends that represent the importance of that person. As an example, from Figure 9 it is possible to conclude that the number of co-occurrences between Abu Bakr Al-Baghdadi and Mullah Omar, that started appearing in July 2014, are stable, while co-occurrences with Ayman Al-Zawairi show a declining trend starting from June 2014.

6. RELATED WORK

This work leverages state of the art engines for semantic (Expert System s.p.a., 2014) analysis and graph analytics (Nguyen, Lenharth, & Pingali, 2013) to extract useful knowledge from textual documents and allow analysts to express complex queries.

Several works in the literature focus on extracting information from documents in natural language. A large corpus of papers refers to sentiment analysis and opinion mining (Pang & Lee, 2008) applied to several information sources. In particular, most of the previous work focus on microblogging platforms (Pak & Paroubek, 2010) (Agarwal, Xie, Vovsha, Rambow, & Passonneau, 2011) (Deng, et al., 2013) and larger documents such as news and blogs (Godbole, Srinivasaiah, & Skiena, 2011).

Our paper relates more closely to works that analyze natural language documents to mine facts and topics and entities (Aggarwal & Zhai, 2012). In particular, several papers focus on the extraction of entities and facts from news articles (Etzioni, et al., 2005) (Pa_ca, Lin, Bigham, Lifchits, & Jain, 2006), but without creating inter-document links. Another class of papers focuses on creating clusters of linked documents that likely refer to the same topics or facts (Wanner, et al., 2014). However, these papers do not consider the notion of time and do not extract facts and entities from clustered documents.

A step further is taken by STORIES (Berendt & Subasic, 2009), a system that clusters documents that contain the same keywords and links them together according to their publication time. However, this work do not leverage state-of-the-art semantic analysis, hence it is not able to extract entities, facts and their relationships from documents. Moreover, it does not support the execution of complex temporal queries similar to those presented in Section 5.

To the best of our knowledge, this is the first paper that presents a processing architecture and a prototype able to analyze large corpora of documents in a scalable way, leverage state-of-the-art engines for semantic analysis, create inter-document links, consider the dimension of time, support complex temporal graph queries and produce several different types of outputs.

7. CONCLUSION

This paper proposes a novel computational architecture that can support analysts in decision-making and sense-making processes, as well as in OSINT activities. The proposed approach takes as its input documents processed by state-of-the-art semantic engines and augments this

information by automatically creating links across different documents and by introducing the notion of time. This framework enables analysts to execute complex queries over large corpora of documents and to highlight how entities, relationships and metrics of interest vary over time. The proposed approach is scalable by design, and experimental results obtained through a prototype demonstrate that the time needed to execute queries is compatible with interactive data analysis. Future work will strive to integrate algorithms for automatic time series analysis in the proposed architecture, thus embedding in the system the ability to identify anomalies, correlations and to produce forecasts for the temporal data of interest.

ACKNOWLEDGMENT

The work presented in this paper has been founded by the EU project "SNAPSHOT: *a Social Network Analysis Platform for the Support of European and Homeland Threat Prevention and Strategies"*. Grant agreement CIPS 4000003757.

REFERENCES

Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. (2011). Sentiment Analysis of Twitter Data. *Proceedings of the Workshop on Language in Social Media (LSM 2011)* (pp. 30-38). Portland, OR, USA: Association for Computational Linguistics.

Aggarwal, C., & Zhai, C. (2012). *Mining text data*. Springer Science & Business Media.

Baldini, N., Neri, F., & Pettoni, M. (2007). A Multilanguage platform for Open Source Intelligence. *8th International Conference on Data, Text and Web Mining and their Business Applications* (pp. 18-20). The New Forest (UK).

BBC News Europe. (2015, 1 3). *Ukraine crisis: Timeline*. Retrieved from www.bbc.com: http://www.bbc.com/news/world-middle-east-26248275

Berendt, B., & Subasic, I. (2009). STORIES in time: a graph-based interface for news tracking and discovery. *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 03* (pp. 531-534). IEEE Computer Society.

Best, C. (2008). Open Source Intelligence. In F. Fogelman-Soulié, *Mining Massive Data Sets for Security: Advances in Data Mining, Search, Social Networks and Text Mining, and Their Applications to Security* (pp. 331-343). IOS Press.

Brockwell, P. J., & Davis, R. A. (2002). *Introduction to time series and forecasting*. Taylor & Francis.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys* (CSUR).

Dean, J., & Ghemawat, S. (2008, January). MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, 51(1), 107-113.

Deng, L., Xu, B., Zhang, L., Han, Y., Zhou, B., & Zou, P. (2013). Tracking the Evolution of Public Concerns in Social Media. *Proceedings of the Fifth International Conference on Internet Multimedia Computing and Service* (pp. 353-357). ACM.

Esling, P., & Agon, C. (2012). Time-series data mining. *ACM Computing Surveys (CSUR)*.

Etzioni, O., Cafarella, M., Downey, D., Popescu, A.-M., Shaked, T., Soderland, S., . . . Yates, A. (2005). Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence, 165*(1), 91-134.

Expert System s.p.a. (2014, 12 13). *Cogito API*. Retrieved from Expert System: http://www.cogitoapi.com/success-stories/intelligence/

Expert System s.p.a. (2015, March 6). *Cogito® Itelligence API live demo*. Retrieved from Cogito® Itelligence API: www.intelligenceapi.com/demo/

Gephi.org. (2014, December 18). *Gephi - The open GraphViz platform*. Retrieved from Gephi: https://gephi.github.io/

Glassman, M., & Kang, M. J. (2012, March). Intelligence in the internet age: The emergence and evolution of Open Source Intelligence (OSINT). *Computers in Human Behavior, 28*(2), 673-682.

Godbole, N., Srinivasaiah, M., & Skiena, S. (2011). Large-Scale Sentiment Analysis for News and Blogs. *Proceedings of the International Conference on Weblogs and Social Media (ICWSM 2011)*. Boulder, Colorado, USA.

Hamilton, J. D. (1994). *Time series analysis*. Princeton: Princeton University Press.

Holt, C. C. (2004). Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, 5-10.

Holzschuher, F., & Peinl, R. (2013). Performance of graph query languages: comparison of cypher, gremlin and native access in neo4j. *Proceedings of the Joint EDBT/ICDT 2013 Workshops*. ACM.

Ihrig, C. J. (2013). JavaScript Object Notation. In C. J. Ihrig, *Pro Node.js for Developers* (pp. 263-270). Apress.

Neri, F., Aliprandi, C., & Camillo, F. (2011). Mining the Web to Monitor the Political Consensus. *Counterterrorism and Open Source Intelligence, Lecture Notes in Social Networks*. Springer-Verlang.

Nguyen, D., Lenharth, A., & Pingali, K. (2013). A lightweight infrastructure for graph analytics. *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*. ACM.

Nisar, M. U., Fard, A., & Miller, J. A. (2013). Techniques for graph analytics on big data. *Proceedings of the 2013 IEEE International Congress on Big Data*, (pp. 255-262).

Odersky, M., Spoon, L., & Venners, B. (2008). *Programming in Scala*. Artima Inc.

Pak, A., & Paroubek, P. (2010). Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *Proceedings of the 7th Language Resources and Evaluation Conference - LREC 2010* (pp. 1320-1326). ELRA.

Pang, B., & Lee, L. (2008). Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval*, 1-135.

Paşca, M., Lin, D., Bigham, J., Lifchits, A., & Jain, A. (2006). Names and similarities on the web: fact extraction in the fast lane. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics* (pp. 809-816). Association for Computational Linguistics.

Richard A. Best, A. C. (2008). Open Source Intelligence (OSINT); Issues for Congress. In T. M. Paulson, *Intelligence Issues and Developments* (pp. 75-97). Nova Science Publishers Inc.

Steele, R. D. (2007). Open source intelligence. In L. K. Johnson, *Handbook of Intelligence Studies* (pp. 129-147). Routledge.

Tausch, N., Philippsen, M., & Adersberger, J. (2011). A Statically Typed Query Language for Property Graphs. *Proceedings of the 15th Symposium on International Database Engineering and Applications* (pp. 219-225). Lisboa, Portugal: ACM.

Tosi, S., Casolari, S., & Colajanni, M. (2013). Data clustering based on correlation analysis applied to highly variable domains. *Computer Networks*, 3025-3038.

Wanner, F., Stoffel, A., Jäckle, D., Kwon, B. C., Weiler, A., & Keim, D. A. (2014). State-of-the-Art Report of Visual Analysis for Event Detection in Text Data Stream. *Proceedings of EuroVis - STARs 2014*. The Eurographics Association.

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010). Spark: cluster computing with working sets. *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing* (pp. 10-17). USENIX.