SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
Fakulta informatiky a informačných technológií

Discovering Entity Relations in Semantic Text Graphs

Michal Laclavík

Habilitačná práca na získanie titulu docent

Habilitation thesis submitted for the Associate Professor degree

February 2013

Acknowledgment

I would like to thank my co-workers from the Institute of Informatics SAS, especially Štefan Dlugolinský, Marek Ciglan, Marcel Kvassay and Martin Šeleng who spent endless hours discussing the topics and tasks presented in the thesis and also greatly helped with setting up the experiments, prototyping, and evaluating the results.

I would like to thank Ladislav Hluchý for providing an excellent working environment, which allowed me to work on the thesis research topics, and Prof. Pavol Návrat for supporting me on this journey.

I would also like to thank Emily Arcuri for proofreading this thesis.

And finally I want to thank my family for their support and my God for his mercy.

CONTENT

ZHRNUTIE

Vyhľadávanie entít alebo objektov začína byť populárnou alternatívou ku klasickému fultextovému vyhľadávaniu dokumentov. Google nedávno spustil vyhľadávaciu službu vyhľadávania entít reálneho sveta, ktorá je založená na štruktúrovaných grafových dátach vytvorených ľuďmi, ako napríklad Wikipédia. Ako riešiť objavovanie a vyhľadávanie entít a ich vzťahov z neštruktúrovaných textových zdrojov je stále jednou z najväčších výziev pre oblasť vyhľadávania a extrakcie informácií.

V práci sa snažíme prispieť k riešeniu tejto výzvy so zameraním na objavovanie vzťahov medzi entitami. Neštruktúrované texty spracovávame pomocou jednoduchých metód pre extrakciu informácií. Následne z extrahovaných entít vytvárame jednoduché sémantické siete, na ktoré aplikujeme metódy z teórie grafov, čo nám umožňuje objavovať vzťahy medzi entitami. Dôležitou časťou je aj interakcia používateľa so sémantickým grafom, čo nám pomáha vylepšiť výsledky extrakcie informácií a následne aj vyhľadávanie relácií medzi entitami.

Objavovanie vzťahov medzi entitami z neštruktúrovaných textov je možné realizovať aj pomocou iných metód, ako napríklad pomocou metód dolovania v textoch. Výhoda prezentovaného prístupu však spočíva vo vytvorení sémantických sietí, ktoré majú podobné vlastnosti ako informačné siete dostupné ako štruktúrované grafové dáta. Obidve grafové štruktúry majú podobné vlastnosti, a preto je možné aplikovať podobné metódy objavovania vzťahov. Navyše pri prepojení takýchto dát by došlo k pozitívnemu efektu jednotného prístupu pre vyhľadávanie v štruktúrovaných a neštruktúrovaných dátach.

Práca obsahuje tiež overenie a vyhodnotenie úspešnosti vyhľadávania vzťahov ako aj vyhodnotenie výkonnosti riešenia. Okrem toho sú opísane viaceré prípadové štúdie, kde sme aplikovali zvolený prístup.

ABSTRACT

Entity Search is becoming a popular alternative for full text search. Recently Google released its entity search based on confirmed, human-generated data such as Wikipedia. In spite of these developments, the task of entity discovery, search, or relation search in unstructured text remains a major challenge in the fields of information retrieval and information extraction.

This thesis tries to address that challenge, focusing specifically on entity relation discovery. This is achieved by processing unstructured text using simple information extraction methods, building lightweight semantic graphs and reusing them for entity relation discovery by applying algorithms from Graph Theory. An important part is also user interaction with semantic graphs, which can significantly improve information extraction results and entity relation search.

Entity relations can be discovered by various text mining methods, but the advantage of the presented method lies in the similarity between the lightweight semantics extracted from a text and the information networks available as structured data. Both graph structures have similar properties and similar relation discovery algorithms can be applied. In addition, we can benefit from the integration of such graph data.

We provide both a relevance and performance evaluations of the approach and showcase it in several use case applications.

1 INTRODUCTION

Entity Search is becoming a popular alternative for full text search. Recently Google released its entity search [ULAN12] based on confirmed, human-generated data such as Wikipedia and Freebase[1], and Facebook is experimenting with graph search over its user generated context. New types of question answering systems such as IBM Watson, based on structured and unstructured data [FERR11], are being developed, but the task of entity discovery, search, or relation search in unstructured text still remains a major challenge in information retrieval and information extraction fields. This thesis attempts to address this challenge, focusing specifically on entity relation discovery.

Entity relation discovery is a topic well covered in literature dealing with structured data where the entities under investigation are already identified. Regarding the data in the form of graphs or networks (semantic or information networks), there are approaches for relation discovery based on the network theory and graph algorithms [CRES97]. This includes methods for ranking relationships in ontologies [ANY05]. When the data is unstructured, text mining methods [CUNN06][PAN06] [CUNN06]such as clustering are usually used for relation discovery [ZH05]. To the best of our knowledge, none of the approaches is suitable for both structured and unstructured data sources.

1.1 Motivation and Background

Graphs or networks often appear as a natural form of data representation in many applications:

- *Social Networks*: these contain a high amount of graph data such as friend networks and information about the interaction among other artifacts like statuses, messages, photos, or tags.
- *Call network*s: networks of communicating people including audio, video or SMS communication with additional data such as location.
- *Internet*: web graph of interconnected web pages.
- *Wikipedia*: the network of Wikipedia concepts including hierarchies, themes or language variations.
- *LinkedData*[2]: a fast growing semantic network data containing metadata about people, geo-locations, publications and other entities.

[1] Graph database with 23 million entities, http://wiki.freebase.com/wiki/Main_Page

[2] http://linkeddata.org/

- *Emails*: social networks are also included in email communication [BIRD06], which can be connected to other objects mentioned in emails such as contact information, people, organizations, documents, links, or time information.

The analysis of email communication allows the extraction of social networks with links to people, organizations, locations, topics or time information. Social Networks included in email archives are becoming increasingly valuable assets in organizations, enterprises, and communities, though to date they have been little explored.

Unstructured text is still the most common medium for information sharing and communication. While it is available on the web, in emails, or within new social media like Facebook, Twitter or LinkedIn, it is also present in enterprises' analytical data like document repositories or even database text fields. All of these web, media communication, and organizational resources preserve a large part of their knowledge in unstructured textual form. In addition, such data is connected with graph/network data through web links, communication links, transactions, or social links and tags (lightweight semantics) in social media and is shared among many users and resources. It has been proved on Web 2.0 (or social web) that the lightweight semantics (tags) and social networks (graph data) give additional value to knowledge sharing, reuse, recommendation and analytics.

The text can be transformed to trees or graph/network structures [LAC11] which have a similar property to the information networks mentioned above. This thesis will further examine those properties.

Searching, analyzing, accessing, and visualizing information and knowledge hidden in such network structures are becoming increasingly important tasks in the area of data analytics [AGA11], but different algorithms must be used for unstructured data processing such as text. In this work we try to create network structures from unstructured text data similar to those of structured data.

Email communication is unique in this respect because it connects social networks (communication) with information networks, which can be extracted from text. We believe that email communication and its links to other organizational as well as public resources (e.g. LinkedData) can be a valuable source of information and knowledge for knowledge management, business intelligence, better enterprise, and personal email search. The future of email [FAU12] is in interconnecting email with other resources, services (like social networks or collaboration tools), or data and entities which are present in email. This was also the main motivation and drive for our work, but we have discovered that the approach could be applied to any unstructured text data, and not only to email communication.

As the size of the real graph data grows, there must be an adequate development in the field of graph data management. Typically, software libraries designed for graph data processing store the whole structure in-memory. This is a serious drawback when the size of the data exceeds the available memory. Recently, there has been a growing interest in managing graph data persistently. Several important research and development directions include:

- *Triple stores*: semantic web databases focused on storing semantics in the form of triples like Virtuoso, Sesame, OWLim or SHARD[3].

[3] http://sourceforge.net/projects/shard-3store/

- *Graph databases* or *graph APIs*: Neo4j, Virtuoso, SGDB[4], or JUNG, which allow graph manipulation, traversing, or persistent data storage.
- *Blueprints*[5]: a common Java API for graph databases, similarly as JDBC for relational databases.

Fast graph traversing is the most important feature when querying large graphs. The challenge is to make the graph querying scalable, since graph traversing has to deal with random access pattern to the nodes [LUM07]. Due to this fact, graph databases try to load most of the data into memory. Scalable processing on parallel, shared-nothing architectures is just emerging, since even big enterprises like Facebook or Google still need to solve large and scalable graph processing. Google has published its Pregel [MAL09] solution for graph batch processing. Similarly, open-source solutions like Hama or Giraph[6] based on the Pregel idea are emerging. However, to the best of our knowledge there is no scalable solution yet for real-time graph querying.

In the presented work we do not yet address the scalability issue, but we are aware that this issue needs to be solved as well. We take performance seriously and evaluate the performance of our approach on large graph networks.

1.2 Text Graphs

The idea of building or extracting graphs from text is not new and it is used to accomplish many tasks in Natural Language Processing (NLP) [MIH11] related to tasks in syntax like Part of Speech (POS) tagging, semantics like word sense disambiguation or applications like topic identification, summarization or machine translation. These topics were also the focus of a series of TextGraphs workshops[7]. In [HASS12], text graphs are used to create signed social network from text discussions. In [MIN12] graph walk, algorithms are used to discover related words. An example is discovering synonyms based on the construction of sentence words graphs [MIN08], where text graphs and random walk algorithms are also used to achieve named entity extraction, message foddering or person name disambiguation. Text graphs are also introduced as a way to enhance concept maps [NUTT04]. In our work we try to use text graphs for entity relation discovery.

1.3 Overview of the Contribution

The main contribution of this thesis is in searching and discovering entity relations in unstructured text using graph data structures extracted from the text. The extracted graph structures have similar properties to other information or social networks, which opens up the possibility for integration of structured and unstructured data. Another advantage compared to other relation discovery approaches comes from user interaction. When the users search for relations, they can interact with the underlying

[4] http://ups.savba.sk/~marek/sgdb.html

[5] https://github.com/tinkerpop/blueprints/wiki/

[6] http://incubator.apache.org/giraph/

[7] http://www.textgraphs.org/

graph data by deleting or merging entities and thus immediately improve search results, specifically discovered relations.

We evaluate relation discovery and showcase possible improvements coming from user interaction. We also evaluate the approach by providing several use cases in which the approach seems to be relevant.

1.4 Thesis Structure

The Thesis is structured as follows: Chapter One provides the introduction, motivation, and background of the research. In Chapter Two we discuss the approach for entity extraction and graph/network creation. We also provide evaluation of the extracted data as well as improvements of the extraction through user interaction with the data. Chapter Two ends with a section describing the network properties of the graph data extracted from various text sources and use cases, which are later described in Chapter Four. Chapter Three presents the relation discovery approach. First, we discuss the problem of fast graph traversing algorithms in large graph structures and in real-time graph querying. Next, the entity relation discovery algorithm based on the spreading activation is discussed, followed by a description of the user interface (gSemSearch) for relation discovery. At the end of Chapter Three we provide both a relevance and performance evaluations of the approach.

1.5 History of the Research

The thesis summarizes and builds on top of the results published over the past few years, which are also listed in the appendix.

The work on text processing, semantic annotation, and information extraction started back in 2006, but it was fulfilled by publication [LAC09] focusing on pattern-based annotation of text resources. This method was further improved and evaluated through email communication in enterprises [LAC11] where we had already explored the idea of semantic trees and networks.

We started to apply algorithms from the graph theory on the networks extracted from email in [LAC10], where we used the spreading activation to infer and evaluate relations between persons and telephone numbers. The results were satisfactory, but we have discovered scalability problems when performing tests on larger datasets which we described in [LAC11B]. We also improved the query user interface and user interaction with graph data. Performance problems were addressed to some extent in [LAC12], where we processed the whole Enron Email Corpus.

The results published in the article [LAC12] summarize most of the work presented in the thesis. However, in the article we focused mainly on email communication, while in the thesis we try to extend, test, and evaluate the results on a wider collection of text data and several semantic graphs extracted from such data. In addition, we also test our approach on information networks from other sources like LinkedData or event graphs.

Several other published papers [DLUG12][LAC11C][LAC12B][LAC12C][TAV12][MOJ12] are related to use cases described in this thesis.

2 EXTRACTING SEMANTIC GRAPH

In this chapter we discuss information extraction techniques focusing on Named Entity Recognition, explain how entities are recognized, and also explain how semantic trees and graphs are constructed. We discuss how information extraction can be improved by user interaction and evaluate our approach. Additionally, we also examine and discuss network properties of the extracted semantic networks.

2.1 Rule-Based Named Entity Recognition

Information Extraction (IE) techniques [CUNN06] usually focus on the five main tasks of information extraction defined by the series of Message Understanding Conferences (MUC):

1. *Named entity recognition* (NE): finding entities. Finds and classifies the names, places, etc.
2. *Co-reference resolution* (CO): aliases and pronouns referencing the entities. Discovers the identity relations between entities.
3. *Template element construction* (TE): properties or attributes of entities. Adds descriptive information to NE results (using CO).
4. *Template relation construction* (TR): relations between entities. Finds relations between NE entities.
5. *Scenario template production* (ST): events involving entities. Fits TE and TR results into specified event scenarios

We described in detail [LAC11] the state-of-the-art in information extraction and advantages of pattern-based information extraction. We will not address it in this thesis. We assume that information extraction techniques are in place and provide us with useful Named Entity recognition (NE – the first IE task). The output is based on key-value pairs representing NE. The work presented in this thesis helps in relation discovery among entities and thus it solves mainly the fourth IE task focused on relations–TR. Since we do not distinguish between NE or TE and all entities are treated as key-value pairs, the results of relation discovery are always related entities to one or multiple entities. This means that discovered related entities can show relations (TR), entity aliases (CO), and entity properties (TE).

For the Information Extraction we use Ontea [LAC09] IE techniques [LAC11], but any other IE tool that provides key-value pairs with position in text can be used. Ontea is based on regular expressions and gazetteers as shown in Figure 1. Applied patterns and

gazetteers extract key-value pairs (*key*: object type; *value*: object value represented by the string) from a text as seen on the left side of Figure 1. If there is textual data present in binary form (e.g. PDF attachment) it is, if possible, converted to text before the information extraction process. Ontea is able to detect document segments such as message replies inside of emails. The extracted key-value pairs are then used to build the tree [LAC11] (right side of Figure 1) and the network of entities [LAC11] as a graph structure (Figure 2).

The Ontea IE tool is able to connect other extraction/annotation tools like GATE[8] [CUNN11], Stanford CoreNLP[9], or WM Wikifier[10]. In most experiments presented in this thesis we have used pure gazetteers and regular expressions, along with some extra rules to support better user interaction. In the example below (Figure 1), the WM Wikifier was used and has annotated *front desk* text as *Receptionist*[11], detected *Executive Director*[12] and also recognized text *north tower* as *List of tenants in One World Trade Center*[13].

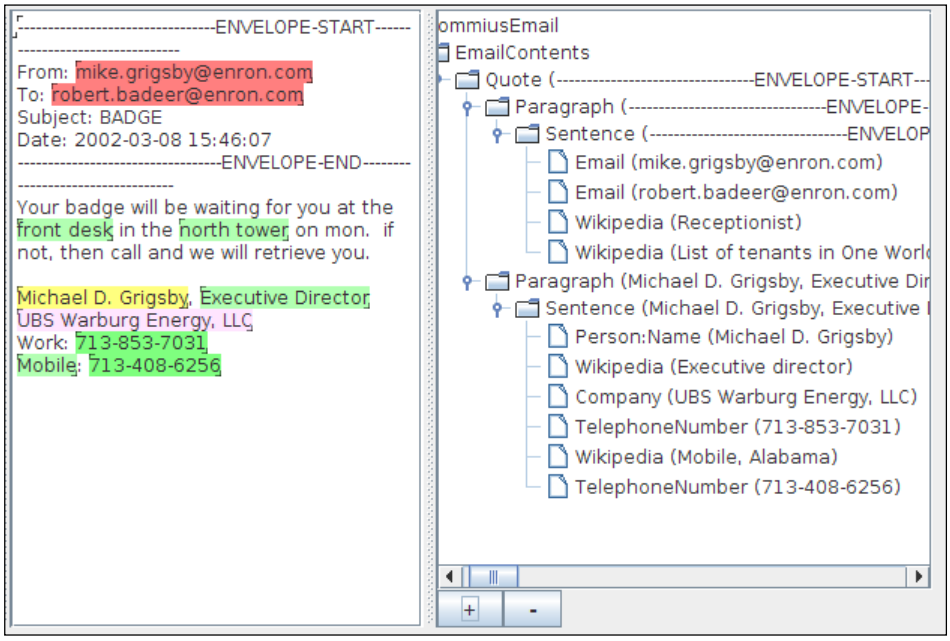


Figure 1: User interface of the IE tool Ontea [LAC11], with highlighted extracted objects (left) and tree structure (right), which is used to build social network graphs (Figure 2)

[8] http://gate.ac.uk/

[9] http://nlp.stanford.edu/software/corenlp.shtml#About

[10] http://www.nzdl.org/wikification/

[11] http://en.wikipedia.org/wiki/Receptionist

[12] http://en.wikipedia.org/wiki/Executive_Director

[13] http://en.wikipedia.org/wiki/List_of_tenants_in_One_World_Trade_Center

2.2 Semantic Trees and Graphs

We can see a graph built from two emails in Figure 2. Such a graph can be built from any text collection in which the document is represented by a document node, its paragraph and sentences nodes, and the documents are interconnected by the nodes representing entities which are present in multiple documents.

Note the two telephone numbers, company name, and person name nodes connected to two different sentence nodes in Figure 2. Even though these entities have been found in both emails, they are presented only once in the graph. In this respect, they are unique. For both nodes and edges, we know also the numeric value of node or edge occurrence in the collection. This can be used as edge or node weight. So far we have not used it in the relation discovery algorithm, but we plan to do so in the future.

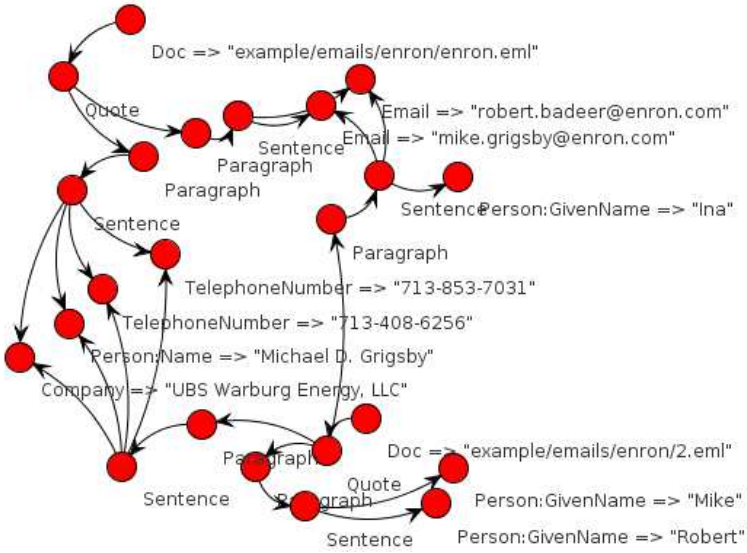


Figure 2: Social Network Graph built from two Enron emails

In Figure 2 you can see the directed graph as it was extracted from text (emails) based on the tree structure, which can be seen in Figure 1. However, when applying graph algorithms (e.g. spreading activation) for relation discovery, we converted the graph to the undirected form. In some cases we had to set some edge types as directed; this was true for LinkedIn and Event graph use cases described in Chapter 4. Thus we defined the graph as directed, but in most cases we considered that if there is an edge in one direction then the edge in the other direction also exists.

Our Semantic Text Graph can be defined as follows: $G = (V, E)$, where V is a set of vertices and E is a set of edges.

$$V = \{v_i, i = 1, n, v_i = (key_i, value_i)\}$$

$$\forall v_i, \forall v_j;\ v_i \in V;\ v_j \in V;\ key_i = key_j \wedge value_i = value_j \Rightarrow v_i = v_j$$

Each vertex v is composed of a $key, value$ pair, where key represents the entity type and $value$ is the text string representing entity (in most cases $value$ is a string extracted from a text document). A unique node is represented by a unique $key, value$

pair. If the same $key, value$ pair is detected multiple times in a document or in multiple documents, they are merged into a single node and connected by edges to the documents, sentences, or paragraphs where they were discovered.

$$e_{ij} \in E;\ e_{ij} = \{(v_i, v_j); v_i \in V;\ v_j \in V\}$$

$$e_{ij} \in E \Rightarrow e_{ji} \in E$$

A generated graph contains edges, which connect vertices/entities with other vertices representing sentences, paragraphs, or documents where these entities were discovered. Edges are type-less with no defined properties, although in the future we would like to use edge weights and edge labels (types of relations), or an edge timestamp for better relation discovery. In our current algorithms the edges are used only to retrieve vertices' neighbors. Text graphs are generated as directed graphs, but in most cases we work with undirected graphs.

2.3 User Interaction with Semantics

In section 2.1 we have described the Rule-Based Named Entity extraction approach. This thesis does not provide any significant contribution in information extraction. The approach used is quite simple, but we kept open the possibility of plugging in new state-of-the-art Named Entity Recognition (NER) systems. The main contribution lies in:

- Simplicity: regular expression patterns and gazetteers
- Configurability: possibility to plug state-of-the-art NER systems
- Tree and graph structure formation from the text

In this section we describe how user interaction can help to improve information extraction and underlying semantic graph. The main results presented in this section were published in [LAC12B].

The user input has been previously considered in the form of tags, LinkedData such as DBPedia [MIL08], or direct user interaction for improving Information Extraction [DIN03]. In our work we use a bit of a different approach. We describe our approach in the DSK[14] use case, in which we crawled about 100 web pages relevant for the DSK case and tested the approach. In the experiment, we simply worked with small graphs extracted from three Wikipedia pages related to DSK, IMF[15] and PS[16] for simplicity.

Patterns and gazetteers can be tuned quite well for applications in which the data is well known as we have shown in [LAC11]. However, it is hard to extract valuable entities from text in open domains such as the Web. We have extracted text graphs from the Web (Section 4.2) by using the simple approach of discovering candidates for named entities. We have simply extracted all words starting with capital letters, which were considered as type-less named entities (marked as NE in Figure 3 and Figure 4). In addition to type-less entities, we were extracting people's names, cities, countries, or dates using the simple gazetteer and rule-based approach. You can imagine that the

[14] http://en.wikipedia.org/wiki/Dominique_Strauss-Kahn

[15] http://en.wikipedia.org/wiki/International_Monetary_Fund

[16] http://en.wikipedia.org/wiki/Socialist_Party_(France)

type-less entities also included many nonsense entities. Here we discuss how they can be deleted or modified by interactively involving users while performing the analytical task of relations discovery.

After extracting text graphs from the web pages, we received many false entities related to *IMF* or *Straus-Kahn* as you can see in Figure 3 and Figure 4.

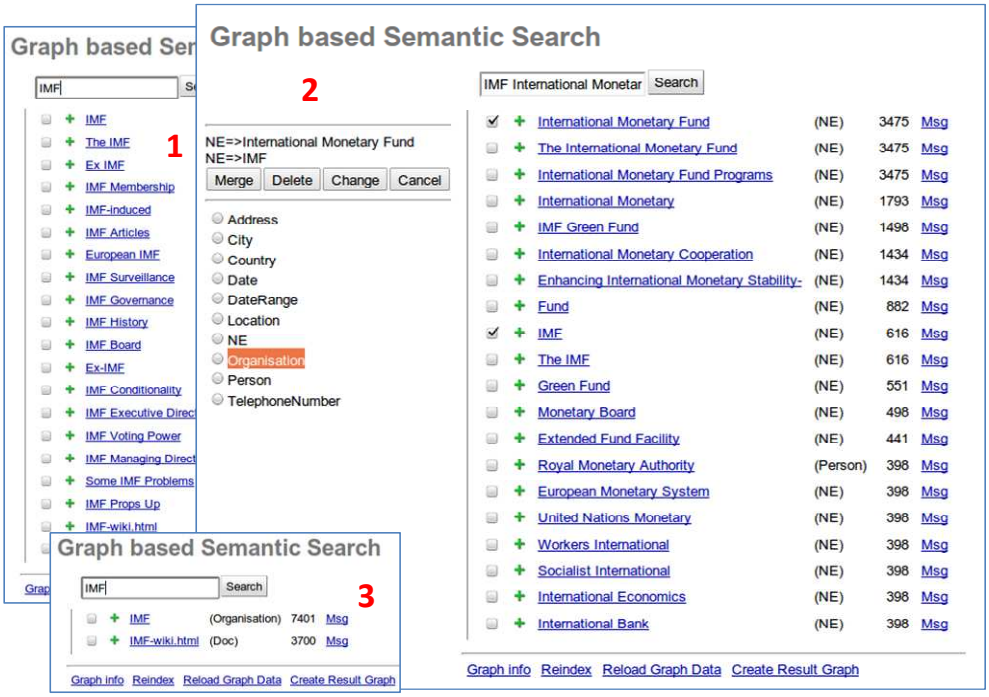


Figure 3: Full text search and user interaction with IMF related entities

Figure 3 shows us full text search results for *IMF* on Screen 1. On Screen 2, we have selected type-less entities *IMF* and *International Monetary Fund* and changed their type to *Organization*. This had an impact on creating positive gazetteers, where both organization names were added to the list of organizations. When the collection was reparsed, *IMF*-related, wrongly identified entities such as *Ex-IMF*, *The IMF*, or *IMF Voting Power* (Screen 1), disappeared from the graph (see Screen 3) based on defined rule, where type-less entity cannot overlap confirmed or type-defined entity.

Similarly in Figure 4, we can see *Strauss-Kahn* as a type-less entity where other type-less entities such as *Strauss-Kahn Case or Strauss-Kahn Released After* were detected (Screen 1), but after modifying the entity type to person and reparsing, only relevant *Person* entities are present in the graph data.

An important feature for user interaction is deleting wrongly identified entities. In our case, we get rid of many such entities by redefining the entity type. However, simple approaches can detect many one-word entities such as days of the week and common words with capital letters in article titles, menus, or at the beginning of sentences. Such ubiquitous entities can then interconnect the data which does not belong together. Even if we identify some entities correctly, sometimes is better to delete them to accomplish the intended analytical task. For example, if *BBC* is detected in each web page from a *BBC* crawl application, it makes sense to remove the *BBC* entity since it connects all of

the data and adds no information. When deleting the node, the entity node with all of its related links (edges) is deleted. Thus, any user interaction has immediate impact on the quality of search results. When the user deletes the node, negative gazetteers are being created. After reparsing the data, all such nodes should disappear from all of the documents where they were previously detected.

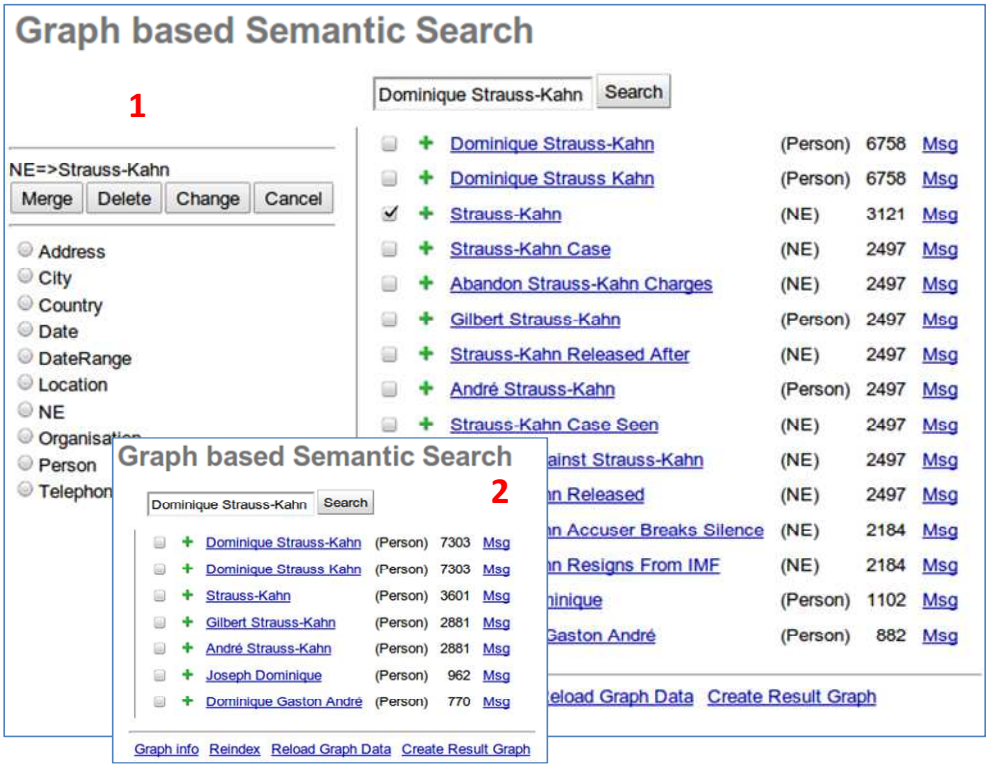


Figure 4: Full text search and user interaction with Straus-Kahn related entities

Names of cities are detected by checking with a list of major cities in the world. This list, however, also includes *Mobile*, a city located in USA. Therefore, when parsing technical news, *Mobile* will be wrongly detected as a city in many places. It makes sense to delete it, thus turning it into a negative gazetteer. A similar case presents itself with human name recognition. People's names are detected based on first name lists and regular expressions. This simple approach works quite well until it detects nonexistent people like *Ivory Coast*, *Roman Catholic Church*, or *Royal Monetary Authority* (see Figure 3, screen 2) because *Ivory*, *Roman*, and *Royal* are valid personal first names. It is easier to delete wrongly identified people or change the entity type than to discard useful detection rules.

As we mentioned earlier, we have experimented with this approach using the web data from BBC, Wikipedia, and the web pages found by Google API on concrete topics. In this same way, we have also gathered web pages on the DSK use case. In order to keep things simple, we have provided an evaluation of only three pages from Wikipedia.

Table 1: Entity and Graph data changes statistics.

	Raw Occurrence	Clean Occurrence	Diff	Diff %	Raw Count	Clean Count	Diff	Diff %
Entities	4682	4776	94	2.01%	2881	2857	-24	-0.83%
Edges	4678	4772	94	2.01%	4383	4377	-6	-0.14%
Person	704	841	137	19.46%	409	411	2	0.49%
Organization	0	360	360		0	4	4	
NE	1893	1490	-403	-21.29%	918	860	-58	-6.32%
Total	2597	2691	94	3.62%	1327	1275	-52	-3.92%

Table 1 compares the statistics of the extracted entities from the *raw* data (i.e. before the user interaction with the graph) with those of the *clean* data (after some cleaning operations done by the user). The difference between the *occurrence* and the *count* columns is that in *occurrence* we count multiple occurrences of the same entity as detected in the text, while in the *count* column every key-value pair (e.g. *Organization=>IMF*) is counted only once. This is because it appears only once in the graph, however Edges represents occurrence in sentences, paragraphs, or documents.

The user interaction captured in Table 1 involved 27 user operations (taking about 2-3 minutes of browsing and interacting with the data) and brought about many changes in the further extraction process. For instance, it resulted in 900 changes (32 times more) in text and 64 changes (1.8 times more) in the graph nodes/entities, as shown in Table 2. Overall, the beneficial effect far outweighed the spent user time. The user immediately experiences the improved results and, after reparsing the data based on new positive and negative gazetteers, the number of changes (*clean* columns) further increases.

Table 2: Entity and Graph data changes statistics

Changes Occurrence	900
Changes Graph	64
Deleted Entities	21
Change Type Entities	6
Operations together	**27**
Improvement occ. count	873
Improvement occ.	32.33x
Improvement count	37
Improvement	**1.76x**

Table 2 compares the number of changes done by the user to the resulting number of changes in the graph data. We see that in the final graph this translates into roughly 1.8 times more node/entity changes,most of them leading to better search results.

User involvement is perceived as quite time consuming. However, if the users see the immediate impact in the form of better search results and better identification of the entities in text, they may be willing to do it. We would like to further extend the approach to an automatic creation of training data sets for IE machine learning approach.

2.4 Evaluation of Extracted Semantics

We have evaluated the rule-based information extraction approach in [LAC11]. The focus was on extracting personal names, postal addresses, and telephone numbers from emails. Therefore, we manually annotated these kinds of named entities in each of the 50 Spanish emails from the Commius project, to have a so-called golden standard for

evaluation. We evaluated the extraction results automatically by comparing them to manual annotations. We considered the information extraction result to be relevant when it was strictly equal to the corresponding manual annotation. This means that both the result and the annotation have had exactly the same position in the text. Evaluation results are summarized in the table below.

Table 3: Evaluation of information extraction - strict match

Type	Total relevant	Total extracted	Relevant extracted	Recall [%]	Precision [%]	F1 [%]
Personal name	779	788	499	64.06	63.32	63.69
Telephone number	262	178	166	63.36	93.26	75.45
Fax Number	139	127	121	87.05	95.28	90.98
Postal address	170	134	75	44.12	55.97	49.34

As we can see in Table 3, except for the fax number and telephone number extraction, the results are not high. This is due to the strict comparison of the extraction results against the annotated set during evaluation, during which many good results were rejected. For example, there were many cases in which the extracted name was almost equal to the annotated one, but it differed from the annotation by the prefix "D." Although this prefix could be considered as an abbreviation for a first name, in our case it stood for the Spanish word *Don* (Sir) and therefore was not included in the annotation. During manual annotation, we did not consider the title to be part of a personal name. Postal addresses also suffered from similar drawbacks; they were written in various formats, so many were extracted partially or they were not extracted at all. Partially extracted addresses were considered to be false matches. For example, if only a postal code with a city name was extracted from the whole address, it was considered a false match even though the postal code and city were located correctly. The same problem occurred with telephone numbers. There were many extractions which overlapped the manually annotated telephone numbers by one or more characters.

Therefore we decided to perform another evaluation with looser matching criteria because, in reality, many extracted results are still useful and valid.

Table 4: Evaluation of information extraction - intersect match

Type	Total relevant	Total extracted	Relevant extracted	Recall [%]	Precision [%]	F1 [%]
Personal name	779	788	672	86.26	85.28	85.77
Telephone number	262	178	178	67.94	100.00	80.91
Fax Number	139	127	125	89.93	98.43	93.98
Postal address	170	134	133	78.24	99.25	87.50

In the second evaluation, we considered the information extraction result to be relevant when it intersected with the manual annotation. The results of the evaluation were much better (Table 4), and showed us that if we enhance the information extraction from emails we can get better recall and precision.

This evaluation was completed two years ago, and we did not provide a new evaluation on current datasets presented in the thesis. Good information extraction results have a significant impact on relation discovery. In this thesis we do not focus on this topic but more on the relation discovery itself. However, we believe similar results as presented in Table 4 (success rate between 85%-95%) can be achieved in the discussed datasets with the help of user interaction.

In [LAC11] we have also proved that a skilled information extraction expert can customize extraction rules for the application within a couple of hours with satisfactory results.

In the future we would like to apply the machine learning approach for information extraction, in which the knowledge engineering approach will be combined with machine learning. The developer will define patterns in a similar way as now, but if several patterns match the same string in the text (e.g. a person or a location), the user will need to resolve which one is valid and when. Machine learning can help set up the probabilities of relevance match based on the training data coming from user interaction. The training set will cover positive examples (user annotations or changes of entity type) as well as negative examples (deletions).

2.5 Network Properties of various Graph Datasets

In this chapter we discuss the network properties of semantic graphs extracted from text data. They have similar properties to web graph, social networks, and information networks such as Wikipedia. The similarity of properties is important because it opens new possibilities for adapting similar algorithms and tools for information networks or for combination and integration of graph data from both structured and unstructured data sources, thus applying the same approach to relation discovery and semantic search.

We illustrate details mostly through the graphs extracted from emails (Enron corpus), but we discuss other sources like semantic graphs extracted from web data (BBC, LinkedData, DSK), from a single monolithic document (Gorila), and also from event graph (agent simulation) and LinkedData (ACM publications) which can have different properties than information networks. The analysis of real-world networks has shown that they usually have several common properties, such as power law degree distribution, small-world property, and high clustering coefficient.

Small world networks

Small world networks are graphs in which any two random nodes can be connected by a relatively short path. These networks appear in many applications, but they are also typical for web graph, Wikipedia, or social networks.

Degree distribution

The degree of a node in a network is defined as the number of connections it has to other nodes, while the degree distribution is the probability distribution of these degrees over the whole network. In most cases, small world networks follow the power law degree distribution [STR01]. When this degree distribution is shown on the log scale, it can be interpolated by a linear function. It also forms a so-called long tail. In Figure 6 we show the degree distribution of DBPedia and the networks with which we have experimented.

The power law degree distribution is a probability distribution that behaves as follows:

$$p(x) \sim cx^{-\alpha}$$

When using a log scale on both axes, we can interpolate it by a linear function with the coefficient $-\alpha$.

Complementary cumulative distribution function (CCDF)

In addition to degree distribution, we can also compute CCDF. In CCDF we sum all degrees of nodes with the degree higher than a given degree (x). The advantage is that a long tail is transformed into a curve/line that can be more easily interpolated by a linear function. Figure 8 shows CCDF on the graphs with which we have experimented. CCDF can be also interpolated by a linear function when using log scale on both axes, but the coefficient of the linear function is $-\alpha + 1$.

The formula for CCDF is as follows:

$$\mathrm{P}(x) = P(X > x)$$

Clustering coefficient

This is a property of graphs or networks which describe how much the nodes tend to interconnect to each other. There are three different measures for this coefficient: local, global, and the average clustering coefficient. The local clustering coefficient is a measure that expresses what proportion of the node's neighbors are also direct neighbors. It is done by measuring the number of existing edges between the neighbors of a node. The clustering coefficient is equal to 1 when the node neighbors form a clique. Local clustering coefficient of a node v_i, where N_i is a set of v_i neighbours, with out-degree k_i is:

$$C(v_i) = \frac{|e_{jk}: v_j, v_k \in N_i, e_{jk} \in E|}{k_i(k_i - 1)}$$

Usually, when talking about a clustering coefficient we refer to the average clustering coefficient, which is calculated as the average of the local ones for all the nodes in the graph.

Assortativity coefficient

Degree assortativity coefficient (AC) denotes a tendency of nodes to be connected with other nodes of a similar degree. It is defined as the Pearson correlation coefficient of the degrees of pairs of nodes connected by an edge in the network [NEW03]. If M is the number of edges and j_i and k_i are the degrees of the i-th edge, the assortativity coefficient can be computed as follows:

$$r = \frac{M^{-1}\sum_i j_i k_i - [M^{-1}\sum_i (j_i + k_i)/2]^2}{M^{-1}\sum_i (j_i^2 + j_i^2) - [M^{-1}\sum_i (j_i + k_i)/2]^2}$$

According to [NEW03], social networks tend to have a positive assortativity coefficient so the networks are assortative, while networks such as internet, biological networks or other information network tend to have negative assortative coefficient and we refer to them as being disassortative. For directed networks, we can distinguish in-degree (number of edges oriented towards the nodes) and out-degree (number of edges oriented from the node). In our study we have computed assortativity of the graphs we have experimented with.

2.5.1 Email Graph Corpus

We have tested the email graph approach on several email datasets. Here we present the results from Enron Email Corpus [KLIMT04]. The Enron Corpus was analyzed in many ways including social (communication) networks [CHAP05] and its visualization. We have created an Enron Email Graph, which contains various entities and social networks extracted from the emails. Each processed email has its own node in the graph with connections to named entities (NE) extracted from this email such as people, email addresses, and telephone numbers. Named entities of the same type and value (e.g. "Person" => "John") are unique in the graph, so one entity found in a different emails is presented only once in the graph, but it is connected to all the emails from which it has been extracted.

Extracted entities

The resulting graph contained 8.3 millions vertices and 20 million edges extracted from half a million messages. The resulting graph file was created in the form of a text file where each node or edge of the graph is represented by one line in the file.

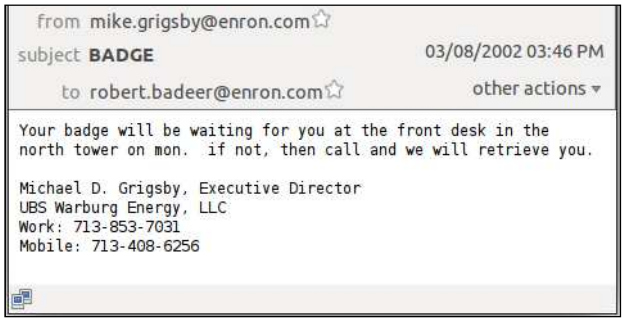


Figure 5: Example of Enron Email

In Figure 5 we can see one email from the Enron Corpus with entities like a person, telephone number, and email addresses. This email will be represented by the following nodes and edges:

```
1 Vertex: Doc=>/home/misos/enron/test/6.eml
1 Vertex: Quote=>/6.eml0:1:0
2 Edge: (Doc=>/home/misos/enron/test/6.eml)=>(Quote=>/6.eml0:1:0)
1 Vertex: Paragraph=>/6.eml0:1:0
2 Edge: (Quote=>/6.eml0:1:0)=>(Paragraph=>/6.eml0:1:0)
1 Vertex: Sentence=>/6.eml0:1:0
2 Edge: (Paragraph=>/6.eml0:1:0)=>(Sentence=>/6.eml0:1:0)
1 Vertex: DateTime=>Fri, 8 Mar 2002 06:46:07 -0800 (PST)
2 Edge: (Sentence=>/6.eml0:1:0)=>(DateTime=>Fri, 8 Mar 2002 06:46:07 -0800 (PST))
```

```
1 Vertex: Email=>mike.grigsby@enron.com
2 Edge: (Sentence=>/6.eml0:1:0)=>(Email=>mike.grigsby@enron.com)
1 Vertex: Email=>robert.badeer@enron.com
2 Edge: (Sentence=>/6.eml0:1:0)=>(Email=>robert.badeer@enron.com)
1 Vertex: Person:Name=>Grigsby, Mike
2 Edge: (Sentence=>/6.eml0:1:0)=>(Person:Name=>Grigsby, Mike)
1 Vertex: Company=>ENRON
2 Edge: (Sentence=>/6.eml0:1:0)=>(Company=>ENRON)
1 Vertex: Person:Name=>Badeer, Robert
2 Edge: (Sentence=>/6.eml0:1:0)=>(Person:Name=>Badeer, Robert)
1 Vertex: Company=>ENRON
2 Edge: (Sentence=>/6.eml0:1:0)=>(Company=>ENRON)
1 Vertex: Person:GivenName=>Robert
2 Edge: (Sentence=>/6.eml0:1:0)=>(Person:GivenName=>Robert)
1 Vertex: Person:Name=>Badeer, Robert
2 Edge: (Sentence=>/6.eml0:1:0)=>(Person:Name=>Badeer, Robert)
1 Vertex: Paragraph=>/6.eml659:19:0
2 Edge: (Quote=>/6.eml0:1:0)=>(Paragraph=>/6.eml659:19:0)
1 Vertex: Sentence=>/6.eml659:19:0
2 Edge: (Paragraph=>/6.eml659:19:0)=>(Sentence=>/6.eml659:19:0)
1 Vertex: Person:Name=>Michael D. Grigsby
2 Edge: (Sentence=>/6.eml659:19:0)=>(Person:Name=>Michael D. Grigsby)
1 Vertex: Company=>UBS Warburg Energy, LLC
2 Edge: (Sentence=>/6.eml659:19:0)=>(Company=>UBS Warburg Energy, LLC)
1 Vertex: TelephoneNumber=>713-853-7031
2 Edge: (Sentence=>/6.eml659:19:0)=>(TelephoneNumber=>713-853-7031)
1 Vertex: TelephoneNumber=>713-408-6256
2 Edge: (Sentence=>/6.eml659:19:0)=>(TelephoneNumber=>713-408-6256)
```

Such representation enables relation discovery as presented in the next chapter. In Table 5, we summarize Enron Graph Corpus properties such as the number of nodes of concrete type and the size of the corpus.

Table 5: Enron Graph Corpus data summary

Description	Size/Count	Description	Count
Corpus Size	2.5 GB	Email	162,754
Compressed Corpus Size	217 MB	MoneyAmount	28,992
Messages	517,377	Paragraph	2,631,292
Nodes	8,269,278	Person	167,613
Edges	20,383,709	Quote	533,007
Address	4,997	Sentence	3,800,504
CityName	1,550	TelephoneNumber	26,013
Company	52,286	WebAddress	105,610
DateTime	228,175		

Enron graph properties

Extracted graph has the properties of small world information networks similar to Wikipedia or web graph. In Figure 6 (top right) we can see the distribution of the node degree with the power law distribution coefficient 1.9 computed according to [CLA09]. Regarding other properties, the Assortativity coefficient [NEW03] was negative (-0.02), denoting that the network is disassortative and similar to other information networks. The average local clustering coefficient of the network is high (0.29), meaning that 29% of nodes in a graph tend to cluster together. The average shortest path on a sample of graph data is 6.58 hops.

2.5.2 Text Graphs properties

In this section we examine and discuss properties of several information networks.

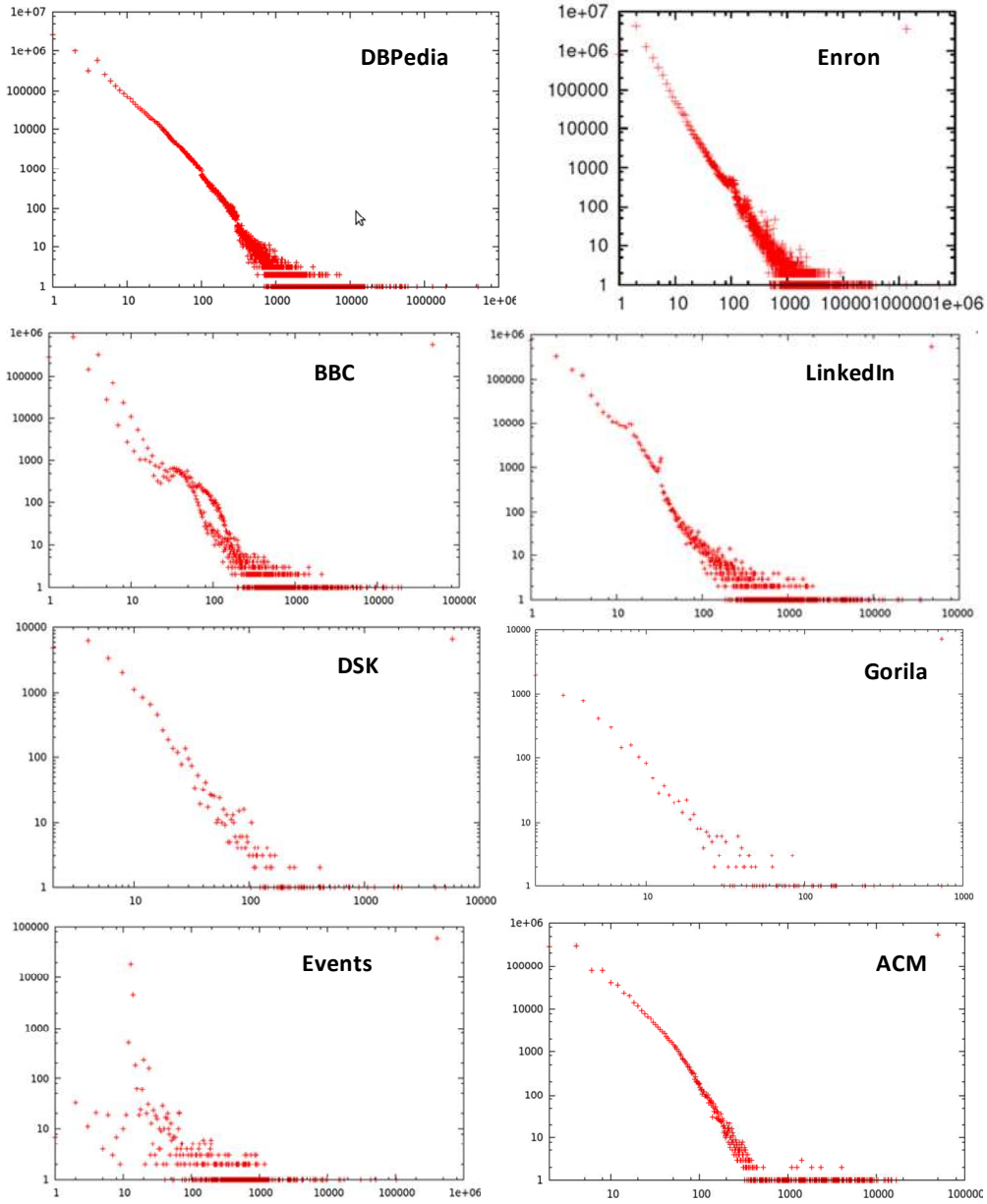


Figure 6: Log scale degree distribution on various graphs

In Figure 6 we provide the probability degree distribution for eight datasets. The node degree is on the x axis and the number of nodes with such a degree is on y axis. The first dataset is the degree distribution of DBPedia. The other seven datasets were used in our experiments. We did not experiment with DBPedia graph, which has similar properties as some of the graphs with which we have experimented, but we do provide it as a typical representative of information networks. For the Enron, Gorila, and DSK

graphs we see clear power-law degree distributions. LinkedIn can be also considered a power-law. For BBC, we notice a strange curve and also something like two independent datasets. We did not investigate this further, but it could be caused by processing BBC news as well as BBC country profiles. For the country profiles, we have extracted and identified more entities so that the topology of trees/graphs extracted from the country profiles is a bit different from the topology of graphs extracted from the news pages. When applying CCDF on BBC (Figure 7) we can see that the power-law distribution is quite valid, especially if nodes with too high or too low of a degree of distribution are not considered.

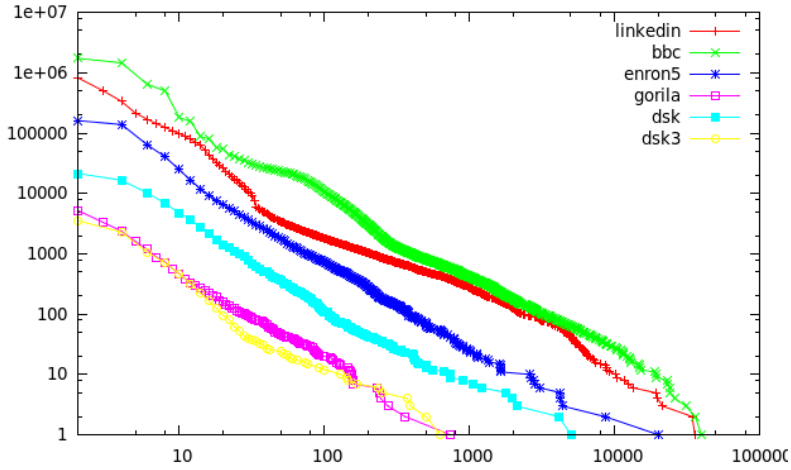


Figure 7: CCDF applied on degree distribution for text graphs from experiments

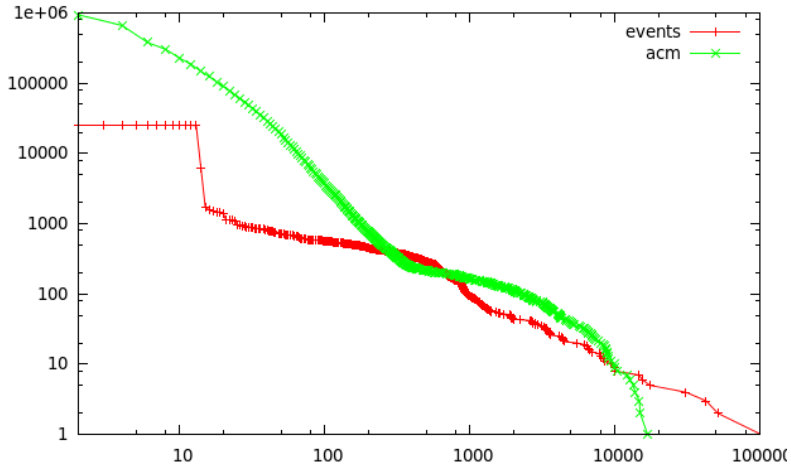


Figure 8: CCDF applied on degree distribution for non-text graphs from experiments

Another dataset is the events dataset obtained from a log file of a multi-agent simulation. The degree distribution of the event dataset is not power-law, as we can also see in Figure 8. In addition we experimented with ACM LinkedData dataset[17]. It has similar properties to text graphs (Table 6), but it seems that the ACM dataset does

[17] http://datahub.io/dataset/rkb-explorer-acm

not have the power-law degree distribution especially when considering its CCDF in Figure 8. This is caused by the strange structure of the long tail seen in Figure 6, where we have quite a high number of hub nodes with various high degrees but always only one node for a wide range of the highest degrees. We experimented by deleting the nodes with a degree above 1,000 neighbors, and such a truncated network indeed behaved as a power-law one.

Table 6: Properties of graphs used in experimentation

Graph/Experiment Name	Nodes	Edges	Average clustering coefficient	Assortativity coefficient	Average shortest path
Enron Full	8,269,278	20,383,709	0.29	-0.02	6.58
Enron5 (5 mailboxes)	160,387	630,330	0.30	-0.04	6.64
LinkedIn	1,564,698	6,094,634	0.36	0.13	6.48
BBC	1,725,900	6,839,358	0.34	-0.05	7.55
DSK	21,518	98,952	0.31	0.39	5.79
DSK3 (3 Wikipedia pages)	2,857	8,754	0.36	-0.14	5.46
Gorila	5,959	23,724	0.31	0.03	6.25
Events (Agent simulation)	25,478	539,328	0.38	-0.25	2.47
ACM Linked Data	941,322	2,198,001	0.34	-0.06	7.30

In Table 6 we provided the properties of graphs used in our experiments. We can see that all of the graphs have a high clustering coefficient of about 30%. The average shortest path was computed for just a sample of random nodes, and it was between 5.5 to 7.5 hops. Events graph has very different properties concerning the average shortest path and the degree distribution. For the ACM dataset, we can see that most of its properties are similar to our text graphs. We have also computed the assortativity coefficient. Information networks should be disassortative (with a negative assortativity coefficient) [NEW03]. Our hypothesis was that our relation search algorithm will work better for disassortative networks since we had problems with relation discovery in LinkedIn and Gorila networks which have positive assortativity coefficients. On the other hand, we have noted that the DSK network is assortative with quite a high assortativity coefficient and our relation discovery worked well there. It would then seem that assortativity does not have influence on the results, but more experiments are needed to confirm it.

To conclude, in this section we examined the network properties of our generated text graphs. We could see that the extracted text graphs were not random graphs but had properties similar to other information networks known from various studies and exploited in applications. Since the properties of the networks were similar, we believe that the described relation discovery algorithm can be applied on both types of networks – those generated from unstructured texts, as well as the natively structured information networks. Moreover, it should also be applicable on the networks created by the integration of both kinds of data sources – structured and unstructured.

3 DISCOVERING ENTITY RELATIONS

In the previous chapter we have described both how semantic text graphs are created and the properties they have. It is important to know the graph properties in order to apply search/discovery algorithm on such graphs. We believe much more work can be done on selecting appropriate graph algorithms for entity relation discovery on such graphs, since many algorithms applicable for small world networks can be used. However, in this thesis we describe our experiments in applying spreading activation algorithm and its modifications on the semantic text graphs.

3.1 Spreading Activation Algorithm

The Spreading Activation Method is a common approach for information retrieval [SAL88] in semantic networks [CRES97].

In our approach, we used spreading activation on the graph of a multidimensional social network in a similar way as IBM Galaxy [JUD07], in which the concept of multi-dimensional social network for text processing was introduced. Spreading activation is also used on the Slovak website Foaf.sk [SUCH07][SUCH10] for discovering relations between people and enterprises in the Slovak business register, in recommendation systems [TROUS09], and also in relation discovery in Wikipedia [CIG10]. Spreading activation was also used for semantic desktop search [SCH08].

Additionally, spreading activation was used on big semantic networks [LARKC11] (LinkedData) in the LarKC project, specifically focusing on scalable inference. In general, spreading activation identifies a smaller part of a semantic network for further logic based inference. In [DIX10] the complexity of spreading activation is discussed. In small word networks one can clearly reach all the nodes in a graph within a few spreading activation iterations corresponding to the average shortest path. Thus, as in any graph algorithm, reasonable heuristics reducing the search space is needed.

As we mentioned in the Introduction, random node access is the key problem for fast graph traversing [LUM07], which is also used in the spreading activation algorithm. Simple Graph Database SGDB [CIG10B] was developed to be optimized for spreading activation. SGDB stores information about nodes and edges in an optimized form of key-value pairs.

In our previous implementation [LAC11B] we used an in-memory graph with the JUNG graph library, but we could not even load the full Enron Graph Corpus. Currently, we

use SGDB on a single machine and achieve satisfactory results (Section 3.3 discusses the performance evaluation) on the whole Enron Graph Corpus, the biggest network we have experimented with.

To the best of our knowledge, SGDB [CIG10B] is the best graph engine for real-time graph querying [CIG12]. In our future work we would like to go further, creating a scalable graph querying solution on a shared-nothing architecture cluster. One idea about how to scale is to use a distributed key-value store instead of a single machine key-value store. However, making it scalable will need to involve other techniques, and difficulties in communication and caching can arise.

When performing spreading activation we traverse only a part of the whole network, but this part grows quite fast with the depth of search since we deal with small world networks which have short paths between any randomly chosen nodes. After a few levels of activation, the spreading activation algorithm can reach the whole graph if the decay factor is not set properly. Therefore, we still need to optimize the spreading activation (or other relation discovery algorithm) even when a fast traversing infrastructure like SGDB is used. Most of the algorithms use modifications of Breadth First search and thus the depth of search needs to be optimized for each query. We have experimentally discovered that we cannot set up a common level of depth for different node relations discovery in information networks (such as the text graphs described in section 2.5) to achieve both satisfactory relevant result and satisfactory performance because the graph topology is different in each case. One common factor that needs to be dealt with is the high-degree nodes.

In our algorithm, activation is started from a set of nodes ($S = \{v_1, v_2 \dots v_k\}$). The activation value is a constant ($n = 10{,}000$) determined experimentally. It is also a maximum number of visited nodes. Visited nodes are stored in the set *V*, which contains the starting nodes at the beginning ($V = S$). Starting nodes are put into the queue $P = (v_1, v_2 \dots v_k)$. R is a set of nodes with assigned relevance, which is computed as n/k:

$$R = \{(v_1, n/k), (v_2, n/k) \dots (v_k, n/k)\}$$

1. Because we traverse the graph using Breadth First method, when the queue is defined as $P = (p_1, p_2 \dots p_l)$, we first take out the first node for processing $p = p_1; P = (p_2 \dots p_{l-1})$. Then the queue is processed for each p until $P \neq \emptyset \wedge n > 0$. For each p, all of its neighbors are defined as a set N_p:

$$\exists\, e_{pb} = \{(p, b)\} \Rightarrow\ b \in N$$

2. For each $b_i \in N_p$ we compute new relevance value of node $q_b = q_p/|N_p|$. We know the value q_p of node p because $(p, q_p) \in R$. We process the neighbours of p only if $q_b > threshold \wedge n > |N_p|$, otherwise the next node from *P* is processed.
3. Each b_i is added into queue $P = (p_2 \dots p_{l-1}, b_i)$, but only if it does not already belong to the set of visited nodes V. After processing, b_i is added to V.
4. If $(b_i, q) \in R$, then it is replaced by $(b_i, q + q_b) \in R$, otherwise $(b_i, q_b) \in R$.
5. When all b_i are processed, n is decreased by the neighbor count of node p:

$$n = n - |N_p|.$$

6. Then we process the next node $p \in P$ from the queue going back to the first step.

When the algorithm finishes, the set R contains the list of nodes relevant to the set of starting nodes (S) with assigned relevancy values (q_i) including the starting nodes.

$$R = \{(r_1, q_1), (r_2, q_2) \dots (r_n, q_n)\}$$

In our algorithm we also define *OR* and *AND* operations over the starting nodes. *OR* operation is done exactly as we described, starting from multiple nodes. When using *AND* operation, we independently run algorithm for each starting node. For example, if running *AND* for two nodes, we get following results sets:

$$R_1 = \{(r_1, q_1), (r_2, q_2) \dots (r_n, q_n)\};\ R_2 = \{(s_1, g_1), (s_2, g_2) \dots (s_t, g_t)\}$$

In final result set for AND operation we include only those nodes which appeared in both sets, and the relevance value is computed by multiplying relevancies: $q = q_i\, g_j$

$$(r_i, q_i g_j) \in R \iff (r_i, q_i) \in R_1 \wedge (s_j, g_j) \in R_2 \wedge r_i = s_j$$

Below we provide the listing of the algorithm written in Java. As mentioned before, we use Breadth First traversing, which is limited to visit only *n* nodes. The algorithm skips the nodes with the higher degree (i.e. higher number of neighbor nodes) than the number of the remaining nodes to be visited. When a node is skipped, we process the next node in the queue. We have experimentally set *n* (the maximum number of nodes to be visited) to 10,000 nodes to have a reasonable search time (around one second) and satisfactory relevant results. The same number *n* is also used as the initial activation value, which is divided by the number of neighbor nodes in the next step. If we have more than one node as activation node, we also divide this initial activation by the number of starting nodes.

```
public Map<Result,Double> relatedBreadthFirst(Set<Entity> startNodes) {
    Map<Result,Double> rM = new HashMap<Result, Double>();
    LinkedList<Entity> rLL = new LinkedList<Entity>();
    int count = visitNodeCount;
    int sizeInit = startNodes.size();
    for (Entity start : startNodes) {
        rLL.addLast(start);
        rM.put(start, (double) count/ (double) sizeInit);
    }
    while (!rLL.isEmpty() && count >= 0) {
        Entity r = rLL.removeFirst();
        visited.add(r);
        int nCount =  g.getNeighborCount(r);
        double v = rM.get(r)/(double)nCount;
        if (v < threshold)
            continue;
        if (nCount<=count) {
            Collection<Entity> rC = g.getNeighbors(r);
            for (Entity entity : rC) {
                if (!visited.contains(entity)) {
                    rLL.addLast(entity);
                }
                visited.add(entity);
                double val = v;
                if (rM.containsKey(entity))
                    val += rM.get(entity);
                rM.put(entity, val);
            }
            count -=nCount;
        }
    }
    return rM;
}
```

We are using Java *LinkedList* as a queue for the nodes to be processed. For each node we simply ask for the number of its neighboring nodes, calculate the activation value accordingly, and decide if we can explore the node or not. In the *count* variable we hold the number of the nodes to be visited, which is decreased by the number of the processed nodes. In future, we should also consider the edge weights and other edge properties.

The algorithm finishes in reasonable times (around one second – based on the setting for the number of visited nodes) and still returns satisfactory relation results, but it can also fail especially if we want to compute the relations for the nodes with high degree. For example, if we would search for relations to the town of *Hudson* or to the state of *Texas*, such entities have too many connections in the Enron Graph Corpus. It does not make sense to infer entities related to *Texas*, but it can make sense to infer entities related to a concrete person as well as *Texas* at the same time. In our current approach, *Texas* would just be ignored. In our future work, we plan to improve our dealing with the high-degree nodes in the sense that we would include them in the relevant results if they were activated from low-degree nodes, but we would not let the high-degree nodes fire and pass their activation value to other nodes (otherwise the whole graph might get included in the results).

3.2 Graph based Semantic Search – gSemSearch

In this section we describe our user interface called gSemSearch, which calls spread of activation search algorithm described in the previous section. The user interface along with some of its features was already discussed in section 2.3. Here we mainly discuss the search features. The text of this section is based on [LAC12].

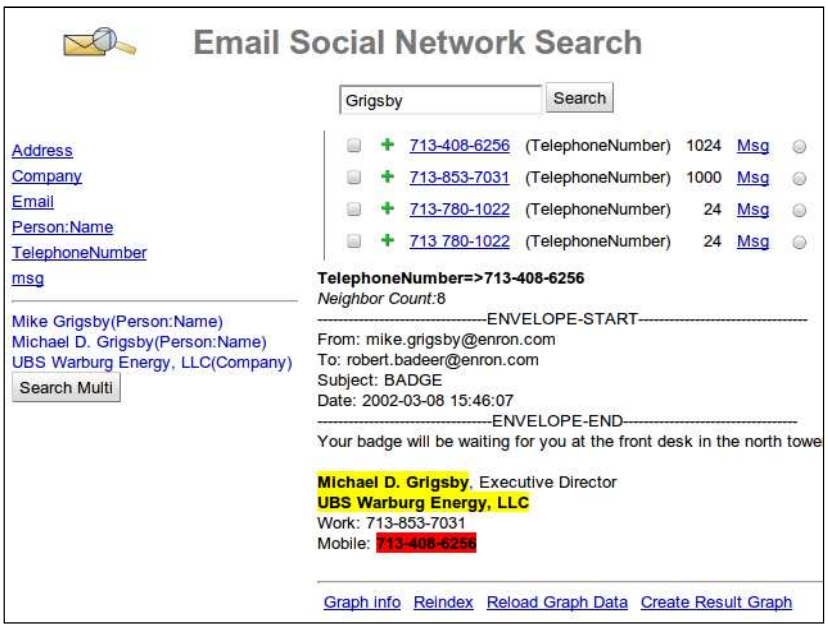


Figure 9: gSemSearch user interface.

To sum things up, the gSemSearch functionality and its user interface allow relation discovery, through which a user can perform a full-text search (e.g., *Gr***by* surname in Figure 9), select starting nodes (e.g., two variations of person names of *Michael Gr***by* and *UBS* company in Figure 9 on the left), and search for the related nodes. A list of nodes with mixed type is returned. It can be restricted to one node type by clicking on the selected type (e.g. *TelephoneNumber* in Figure 9 on the left). This will return nodes of the desired type as seen in Figure 9 (related phone numbers). Our prototype suggests that the starting nodes and the return results are related but does not suggest the type of relation. The type of the relation can be gauged by the user by clicking on the *Msg* links next to the result nodes in the list. This will highlight starting nodes in the most relevant email message in yellow and the selected node in red (note that same objects can be present in multiple messages), which can also be seen in Figure 9.

The search algorithm can also be improved by allowing the users to delete the wrongly extracted objects or to connect various aliases of the same object (e.g. the same company or personal names spelled differently) as seen in Figure 10. Such user feedback enables the search algorithm to learn and return better results in the future. For example, if we merge the three selected person name aliases seen in Figure 10, there will be better results (e.g. phone number, address or organization) returned for any of the aliases.

Figure 10: Prototype GUI with the results of full-text search and several objects (aliases) selected for possible merge or delete. When these object aliases are merged, subsequent searches return better results for any of them.

In addition, the gSemSearch user interface supports actions like node merging and deleting or changing the node type.

We also use a unique approach for synonymy and polysemy of the explored entities (ambiguity and disambiguation). If an entity is represented by more than one node (multiple aliases, similar to the person in Figure 9 or Figure 10), we can use two methods to explore the entities related to such an entity. We can either select all of the aliases and search for the nodes related to this node cluster (Figure 9), or we can merge the aliases to a single node (Figure 10) and explore its relations as if it were a single node. Both approaches have some drawbacks, but satisfactory results can be achieved by their combination. Another problem arises when the same string represents two different entities. We do not provide automatic disambiguation during the extraction,

so two different people with same name will be presented as one node in the graph. However, if some extra auxiliary information is known about the nodes, for example an address or company related to the person, the person node can be selected along with these related nodes, and the search can be performed for other entities related to this multiple selection. This way the sub-graphs related to the other person represented by the same named entity will either not be explored at all, or will be explored/activated only partially.

We have also tested the gSemSearch relation discovery on other data types like graphs extracted from BBC news, LinkedIn job offers, and event graphs of agent-based simulations, so we see the possibility of exploring our relation discovery approach and user interface in other domains in which data can be represented by graph/network structures with properties of information networks. This will be discussed in chapter 4.

3.3 Relevance Evaluation

In this section we evaluate the relevance of the proposed algorithm on various datasets using the information retrieval measures of Precision and Recall.

3.3.1 Evaluation of the prototypes on a set of English emails

In the scope of the Commius project, we have evaluated our approach on the relations between telephone numbers and organizations/people [KVA10].

We created a test email archive consisting of 28 representative sample messages supplied by our partners in Commius. We then ran the prototype with the task to assign telephone numbers to people and organizations, represented by their proper names, which were found in the emails.

Though our primary goal was to evaluate the spreading activation algorithm, we could not completely insulate it from the effects caused by the Information Extractor. These are seen primarily in the figures for Recall. Out of 17 unique relevant telephone numbers in the test emails, the IE identified 13. These 13 numbers were then passed on to the spreading activation algorithm.

The results of the prototype evaluation are presented in Table 7. The relations were identified with a Precision of 76.9% and a Recall of 58.8%.

Table 7: Quantitative evaluation of the attenuated variant of spreading activation

Total relevant	Total found	Correctly assigned	Wrongly assigned	Recall [%]	Precision [%]
17	13	10	3	58.8	76.9

The Information Extraction had quite a large impact on the results since many wrongly identified people names were present in the graph, as well as on the spreading activation algorithm by inferring the relations. Even with this drawback, the Precision results were quite satisfactory. In the next experiment we try to examine the error caused by the imperfect information extraction.

3.3.2 Evaluation of the prototypes on a set of Spanish emails

In the scope of the Commius project, we also tested the algorithm on a set of 50 Spanish emails [LAC10]. The Information Extraction part of the evaluation is described in section 2.4. The testing task again consisted of assigning phone numbers to various persons. For both the personal names and phone numbers, our prototype depended on the information extractor (IE). Its only responsibility was to correctly assign the received phone numbers to the received personal names. This difference in the task (compared to IE) necessitated a different evaluation methodology.

While the extraction of telephone numbers was fairly straightforward, a number of problems resurfaced during the extraction of personal names. Spanish personal names often consisted of four components, which the writers seldom spelled out in full. They might use only three components or sometimes just two. On top of that, they would sometimes write their names with properly accentuation, but at other times they would forgo the use of accents. We therefore decided to accept any extracted string consisting of at least two correctly spelled name components (accents were not required) as a valid variant of a personal name. The telephone number was considered correctly assigned if any of the acceptable variants of its owner's name appeared as the first or second in the list of potential "owners" sorted by score. The results are summarized in the tables below. The number of telephone numbers in Table 8 differs from that in Table 3, because Table 8 represents only the unique telephone numbers for the set of 50 Spanish emails. Moreover, the telephone numbers were re-formatted by leaving only numeric characters in them before they were added into the social graph.

Table 8: Evaluation of the phone number extraction for social network reconstruction

Total relevant	Total found	Relevant found	Recall [%]	Precision [%]
42	32	29	69	90.6

Out of 42 relevant, unique phone numbers in the test corpus (and 32 phone numbers actually extracted), two occurred in the emails which did not contain any personal name (only the company name was present). These two were then irrelevant for the task of pairing phones and personal names as summarized in Table 9. Since there were 40 relevant, unique phone numbers, we expected 40 correct pairs consisting of a phone number and a personal name. This was for us the *Total relevant* for the pairing task. The number of assigned pairs (shown in the *Total Found* column) is now 30, since it excludes the two irrelevant phone numbers. The number of correctly assigned phone numbers to persons is shown in the *Relevant found* column.

Table 9: Quantitative evaluation of the assignment of phones to personal names

Total relevant	Total found	Relevant found	Recall [%]	Precision [%]
40	30	18	45	60

Table 9 implies that out of 30 found pairs, 12 were wrong. Out of these, six were caused by the fact that the Information Extractor failed to extract the name of the person that actually "owned" the phone number. A further three errors were caused by the extraction of wrong (corrupted) phone numbers, as implied by Table 8. This means that out of the 21 pairs which the social network extractor could possibly pair correctly, it only failed to do so three times. This would give a Precision rate of 18/21 = 85.7%, which demonstrates the potential of spreading activation in reconstructing social networks. The immediate conclusion is that we need to focus primarily on improving the quality of the information extraction of text-based data in multilingual contexts.

The 60% Precision of the social network extraction is less than what we obtained with English emails (Precision of 77%), but there are clear possibilities for improvement, especially at the information extraction stage which is the base for next steps. The improvement of information extraction from emails is our primary target, and we have noted several opportunities for the improvement of extraction patterns which will then give much better results. We expect the improvement to be as good as the one shown in Table 4. As already discussed, most of the errors in the social network extractor were due to the lower recall of the preceding information extraction step. Spreading activation can deal with the lower precision of the information extraction but cannot cope with low recall (i.e. when something needed was not extracted). In this respect, the main opportunity for improvement lies in making the partially good results of the information extraction (summarized in Table 4) available for the social network extractor, since now only strict IE results (shown in Table 3) are really exploited for social network reconstruction. These experiments were originally reported in [LAC10], and because of these finding we tried to improve the extraction by extracting more (even possibly false) entities in order to have higher Recall. While this can, of course, lower the Precision, it can subsequently be cleaned by user interaction with the graph.

To conclude, after we performed these experiments we tried to improve information extraction by allowing other state-of-the-art information extraction tools to plug-in, as well as by introducing user interaction with the data. Such interaction can greatly impact the data quality, especially for the nodes activated in the local graph. Wrongly identified information extraction results often appear in search results from the spreading activation. They can be eliminated by deleting some nodes, merging the aliases, or reassigning the entity types as discussed in section 2.3

3.3.3 Precision of the entity relation discovery

In [LAC10] we have evaluated the success (Precision and Recall) of the IE and the success rate of relation discovery (the spreading activation algorithm) with satisfactory results [LAC10]. The discovered relations Precision was 60% for the Spanish email dataset and 77% for the English one. Interestingly, most of the errors were introduced by the imperfect information extraction. When ignoring the information extraction errors, the relation discovery Precision was about 85% [LAC10]. The algorithm tested on small datasets had a reasonable performance (search time) with acceptable results, but when it was applied on larger datasets we discovered the performance problems described in Section 3.4. The algorithm was subsequently optimized for faster performance. It is quite a hard task to evaluate how well the algorithm for relation discovery works on larger datasets. We decided to at least evaluate the Precision of the returned results on BBC dataset. Recall could not be computed since we would have to go through all of the data manually.

In Table 10 we provide a summary of the evaluation experiment on BBC dataset. We have evaluated four types of relations for several queries. First, we have tried to return the list of relevant people for a concrete person (politicians from different countries in this case). When selecting a concrete person such as *Barroso*, we have selected also all possible aliases such as *Mr. Barroso* or *Jose Manuel Barroso*, and then evaluated the returned relevant person list. In the first column we provide the precision rate for the first returned item in the result list. The second column is the precision rate for the first five listed results. In the last column we examined the first five results and if the aliases of the same person appeared (e.g. *Mr. Dzurinda* and *Mikulas Dzurinda* returned for *Mr. Fico*), we considered more than five results grouping aliases together. In the third column we evaluated the Precision of the first five results, but we also considered partially relevant results. For example, in many cases one of the returned names was that of a journalist writing about the country, person, or organization in the query, or people related to the queried entity because of some events mentioned in the processed news. The Person-to-People Precision was quite high, but we cannot be sure about Recall. In the Country-to-People cases we had a problem detecting people related to *Hungary* or *Poland;* therefore, we did not get very good results since human names were also identified based on gazetteers of first names. We have used first names in English, Spanish, Italian and Slovak.

Table 10: Evaluation on BBC dataset

	P@1	P@5	P@5 partialy	P@5 alias
Barroso	1	1	1	1
Sarcozy	1	1	1	1
Fico	1	0,8	1	0,8
Cameron	1	1	1	0,8
Merkel	1	0,8	0,8	0,4
Tusk	1	0,8	1	0,8
Person->People	*100%*	*90%*	*97%*	*80%*
Nobel Peace Prize	1	1	1	1
IMF	1	1	1	0,8
NATO	0	0,6	1	0,6
EU	0	0	1	0
Lisbon Treaty	1	0,8	0,8	0,8
EC	1	0,6	0,8	0,4
NE->People	*67%*	*67%*	*93%*	*60%*
Slovakia	1	1	1	1
Czech	1	1	1	0,8
Hungary	1	0,2	0,6	0,2
Poland	1	0,33	0,67	0,33
UK	0	0,6	0,8	0,6
Country->People	*80%*	*63%*	*81%*	*59%*
Hungary	1	0,33	0,67	
Poland	1	0,2	0,4	
UK	1	0,4	0,6	
Ukraine	1	0,6	1	
Country->City	*100%*	*38%*	*67%*	
Total	**86%**	**67%**	**86%**	**67%**

The good results were returned for type-less entities (NE) to people relations. We did not have such good numbers for precise relation between entities, but when we examined the partially relevant results the Precision rate was 93%.

Based on the provided evaluation, it is hard to draw any conclusions on the precise relevance of the returned results. From all of our experiments we can conclude that the entity relation search method gives good results in many cases, but it relies on the quality of the extracted named entities. In the BBC dataset, the named entity identification strategy was rather simple, producing many false entities and many type-less named entities; however, with a little user effort to clean the data interactively (as presented in section 2.3) the results can often be substantially improved. This held true for most of the datasets with which we experimented. More serious problems were found when experimenting with LinkedIn and Gorila datasets.

When searching the LinkedIn datasets, we wanted to infer job offers based on entities such as locations or skills which had a high node degree, but the algorithm was not able to search deeply enough. This problem occurred because the activation was stopped after a certain number of visited nodes had been reached. We will have to investigate how to deal with entities/nodes with high degrees, but currently we usually ignore them or the activation is stopped in such nodes. Please see more details in the LinkedIn use case description in Section 4.2. In the Gorila dataset, the main problem was that many entities had many different "name" nodes for the same person due to the rich morphology (inflexions) of the Slovak language. That is why it was hard to select appropriate nodes in the initial search when inferring results for a concrete person. This problem can be solved by improving information extraction so that it groups various morphology forms of the same entity together. More details can be found in Section 4.3.

To conclude, spreading activation is a valuable method for finding relations among entities in information networks, which is confirmed not only by our experiments but also by other relevant work [JUD07], [CIG10], [TROUS09] in the field. Semantic text graphs have similar properties to other networks where spreading activation was tested. The challenges for text graphs are better information extraction and scalability. We tried to achieve better information extraction by applying simple extraction methods and then allowing users to interact and improve the data while searching for relevant results. By user interaction, we not only directly improve the results as well as enable better extraction in the following rounds of search, but also create possible training data sets for machine learning. In the future we intend to investigate various machine learning approaches. Scalability or performance is another challenge when working with large networks, since graph traversing is of an exponential nature. We will discuss this in the next section.

3.4 Performance Evaluation

In this chapter we discuss the performance evaluation of entity relation discovery in extracted networks. The performance problems were discussed in [LAC11B] and solved to some extent in [LAC12]. In this chapter we provide a summary of our findings.

Before examining the network properties of the extracted graphs (see section 2.5), our hypothesis was that the performance (search time) should be stable even with large graphs because we always activate only a small portion of the graph. This was found to be valid only to an extent. One problem is that the created semantic graph has the properties of small world networks. For example, in a similar work performed on the Wikipedia graph [CIG10], only two iterations of spreading activation could be

performed before it would visit too many nodes. In [LAC10] we have used 30 iterations, but in large graphs the impact on performance was too high.

Table 11: Email Social Network Search performance evaluation on four queries and five datasets (datasets are represented as table columns).

Number of Mailboxes	1	5	7	10	15
Number of Emails	3 033	9 939	20 521	36 532	50 845
Number of Verticles	41812	159 776	369 932	608 146	835 025
Number of Edges	98566	380 254	971 929	1 796 403	2 514 031
Processing time (ms)	81 672	430 025	1 199 463	1 948 847	2 680 171
Processing time (minutes)	1	7	20	32	45
One Email processing time	27	43	58	53	53
Person:Name=>Mike Grigsby					
Search Response Time	**144**	**446**	**758**	**1 396**	**1 696**
Results	344	463	494	781	761
Fired	6 363	20 732	19 045	23 466	23 839
Visited	112 280	281 060	476 324	939 642	1 174 400
Visited Unique	18 382	53 772	82 219	145 192	178 829
Search Slowed down x Times	**1**	**3,1**	**5,3**	**9,7**	**11,8**
Fired x Times	**1**	**3,3**	**3,0**	**3,7**	**3,7**
Number of messages x Times	1	3,3	6,8	12,0	16,8
Number of verticles x Times	1	3,8	8,8	14,5	20,0
Number of edges x Times	1	3,9	9,9	18,2	25,5
TelephoneNumber=>713 780-1022					
Search Response Time	**5**	**8**	**8**	**12**	**13**
Results	4	4	4	4	4
Fired	116	150	157	181	183
Visited	6 318	8 776	9 550	13 424	14 710
Visited Unique	698	954	1 059	1 424	1 513
Search Slowed down x Times	**1**	**1,5**	**1,6**	**2,3**	**2,5**
Fired x Times	**1**	**1,3**	**1,4**	**1,6**	**1,6**
Number of messages x Times	1	3,3	6,8	12,0	16,8
Number of verticles x Times	1	3,8	8,8	14,5	20,0
Number of edges x Times	1	3,9	9,9	18,2	25,5
Address=>6201 Meadow Lake, Houston, TX 77057					
Search Response Time	**7**	**14**	**28**	**40**	**59**
Results	23	38	71	91	170
Fired	236	515	701	896	1 546
Visited	8 134	15 571	32 336	40 563	58 571
Visited Unique	1 097	1 952	6 526	8 029	11 295
Search Slowed down x Times	**1**	**2,1**	**4,3**	**6,0**	**8,9**
Fired x Times	**1**	**2,2**	**3,0**	**3,8**	**6,6**
Number of messages x Times	1	3,3	6,8	12,0	16,8
Number of verticles x Times	1	3,8	8,8	14,5	20,0
Number of edges x Times	1	3,9	9,9	18,2	25,5
Email=>ina.rangel@enron.com					
Search Response Time	**106**	**552**	**1 162**	**2 156**	**3 017**
Results	732	1 764	2 668	2 809	2 952
Fired	5 165	16 062	17 629	19 716	20 997
Visited	91 199	369 584	865 300	1 694 065	2 326 867
Visited Unique	13 355	54 987	81 757	134 876	168 955
Search Slowed down x Times	**1**	**5,2**	**11,0**	**20,3**	**28,5**
Fired x Times	**1**	**3,1**	**3,4**	**3,8**	**4,1**
Number of messages x Times	1	3,3	6,8	12,0	16,8
Number of verticles x Times	1	3,8	8,8	14,5	20,0
Number of edges x Times	1	3,9	9,9	18,2	25,5

In the experiment presented in Table 11 [LAC11B], we have determined that the optimal number of iterations was four. This value seems to have little to no impact on

the relevance of the returned results. The algorithm implemented in [LAC11B] seems to visit too many nodes (without firing them) even with four iterations (compare the rows *Fired* and *Visited*); moreover, it visits the same nodes several times (compare the rows *Visited* and *Visited Unique*).

As can be seen in Table 11, we have tested the search response time on five datasets from the Enron Corpus. Datasets contained from 3,000 up to 50,000 emails. The biggest dataset resulted in a graph of 800,000 nodes and 2.5 million edges. Before processing, the graphs were loaded into Memory. We have used the Jung[18] library for graph representation.

Later, we updated the algorithm [LAC12] as described in section 3.1. In Table 12, we present a performance evaluation similar to the evaluation above (Table 11) but on the full Enron Graph Corpus. While in [LAC11B], we tested the algorithm performance with 50,000 messages and less than 1 million nodes, but now the algorithm and infrastructure are capable of scaling up to 500,000 messages and 8 million nodes.

Table 12: Search time evaluation on Enron Graph Dataset for chosen entities (nodes)

Person:Name=>Mike Gri*by**	
Search Response Time (ms)	1, 195
Visited	9,441
Visited Unique	4 ,423
TelephoneNumber=>713 780-12**	
Search Response Time (ms)	378
Visited	4, 092
Visited Unique	2 ,627
Address=>6201 M*ow Lake, Houston, TX 77057**	
Search Response Time (ms)	171
Visited	6 ,188
Visited Unique	4, 078
Email=>ina.ra*l@enron.com**	
Search Response Time (ms)	615
Visited	7, 052
Visited Unique	3, 836

We have performed the same searches as in the previous experiment on the full Enron Graph Database for four different types of objects: person, telephone number, address and email address (as seen in Table 12). The various selected types of entities represent a different topology of related sub-graphs explored during the graph traversal. For example, an email address is usually connected to many nodes directly, while a telephone number or address is connected to just a few sentence nodes. When searching for related nodes, different depths of graph traversing need to be explored for different object types. We achieved this by using the algorithm presented in the thesis. The response time was computed as the average from five searches. As we mentioned earlier, the algorithm visited only *n* nodes while traversing the graph, where *n* was set experimentally to 10,000. Thus we see that the number of visited nodes is less than 10,000, and the number of unique visited nodes is even smaller. The search time is usually lower than 1 second, but it varies (from 171 ms to 1,195 ms in our experiment) based on the cached data of the underlying key-value store infrastructure.

[18] http://jung.sourceforge.net/

4 USE CASES

In this chapter we briefly discuss the use cases in which our approach to entity relation discovery was tested. The structure and information partly overlap with the previous sections, especially with Section 2.5, where we have already reported the properties of the datasets associated with these use cases.

4.1 Email Communication

Our approach to entity relation discovery was started, developed and explored mainly in email communication. Since emails comprise both unstructured text (in message bodies) and a social network of communicating people (in message headers), we have tried to discover a common general method for entity relation search in email archives that would cover both the unstructured text and social networks.

Commius Project

The basis for our approach was the Email Social Network Search[19] [LAC11] prototype which was developed in the scope of the Commis[20] project and extended after the end of this project. In Commius, we have tested the relevance of the approach in enterprise emails, described in [LAC11], section 3.3.1, and section 3.3.3. In [LAC11], available in appendix 6.3, you can also find a screenshot of the first prototype of relation discovery interface (Figure 6 in the paper).

Enron Email Corpus

Later when trying to analyze the Enron [CHAP05] email corpus, whose size and properties are reported in section 2.5.1, we had to deal with the slow performance of the algorithm on large graphs, causing us to modify the algorithms as described in this thesis. Details are also available in [LAC12]. Screenshots of the use case based on the Enron dataset are available in Section 3.2.

Email based Enterprise Search

In this section, we briefly describe enterprise entity search based on email and document analysis, which can help small enterprises achieve their business tasks especially those fulfilling some information need. Proof of the concept implementation was provided within the VENIS[21] project in the InterSoft use case. This particular use

[19] http://ikt.ui.sav.sk/esns/

[20] http://www.commius.eu/

[21] http://www.venis-project.eu/

case is related to the allocation of development resources to various providers and customers at the same time. Customers usually ask large providers of software solutions to fulfill their complex projects. Since the providers often do not have all of the required development resources available, they need to find suitable subcontractors and involve them in the collaboration so as to complete the project for the customer.

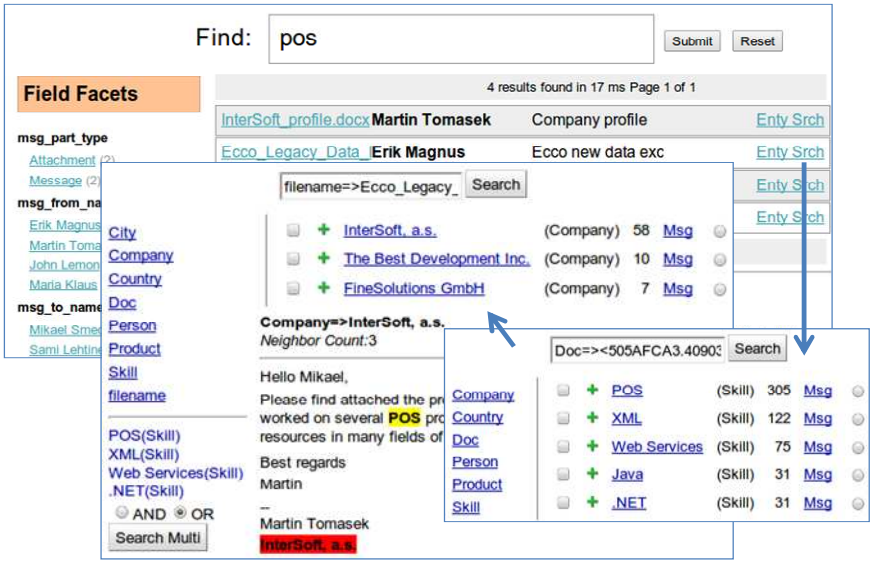


Figure 11: Prototype Enterprise search user interface

In Figure 11 we see an early implementation of search functionality. There is a search field on the top of the screen which can be used to search for documents or emails using full-text search. Full-text search is integrated with an entity search, in which the related entities are displayed by clicking on *Enty Srch* link. The entity search and the recommendation have features as described in the thesis and displayed in Figure 11's two front windows. The front-most one shows the skills detected in the development requirement email, while the one behind it shows the list of companies relevant for the skills. The idea is to return the relevant entities for one or more selected elements (i.e. context) and thus deliver the needed information for the business task represented by the email or document. More details can be found in [LAC12B] and a demo video is also available[22].

4.2 Entity Relation Discovery in Web Documents

In this section we discuss how the entity relation discovery approach can be applied to web documents. We will examine the following three use cases: BBC news, DSK and LinkedIn.

While in the BBC and LinkedIn use cases we have crawled many pages from one website, in the DSK use case we have crawled web pages relevant to one topic returned by Google search.

[22] http://youtu.be/MSS3t__GLdk

BBC news

For the BBC use case, we crawled the BBC news portal. We crawled and processed about 19,000 news articles, in which we then identified about 100,000 entities such as people, organizations, countries, cities, or type-less named entities.

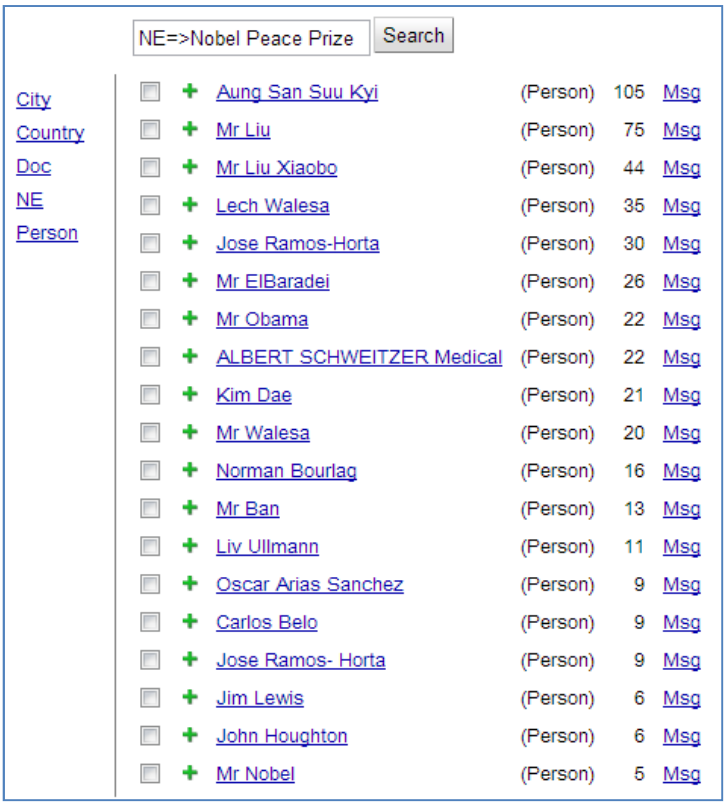


Figure 12: People relevant to the Nobel Peace Prize in the BBC Use case

In Figure 11, we have the list of relevant people returned for the type-less entity *Nobel Peace Prize*. As you can see, the return results are people somehow related to the Nobel Peace Prize. The top of the list clearly contains the prize winners. By the end of the list, though, other people start to appear such as members of the Nobel Peace Prize award committee or those involved in the award ceremony.

Similarly, we can discovery other relations such as people relevant to a person, cities relevant to a country, people relevant to a specific country, or any other interesting relation to one or multiple entities.

In Figure 13 we can see the results returned for two selected entities: *Nato* and *Georgia*. Here we demonstrate the operations OR and AND. On left side, the results of the OR operation are returned. As you can see, *South Ossetia*, *Abkazia*, *Tripoli*, and *Gaddafi* are returned in the list. In an OR operation, highly related results for just one of the entities are returned. In this case, the first two mentioned are relevant to *Georgia*, while the last two are related to the military operation in *Lybia* undertaken by *Nato* at the time of our crawl.

Figure 13: Related entities returned for NATO and Georgia

On the right side of Figure 13, we see the results for the AND operation which more reliably returns the entities relevant for both the input entities. The same holds true for Figure 14, in which we see the results restricted to relevant cities. We can see the chosen AND/OR operation in the bottom left corner of each user interface frame.

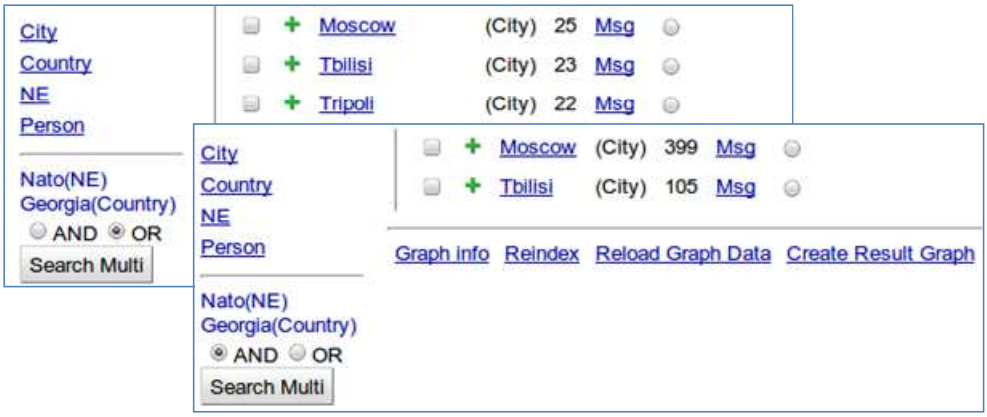


Figure 14: Related cities returned for NATO and Georgia

DSK use case

The DSK use case is related to the Dominique Strauss-Kahn case, involving a French politician and former director of the International Monetary Fund. As we already mentioned in Section 2.3, for this case we crawled about 100 web pages relevant for the DSK case and tested the approach. The selected 100 pages were the first 100 Google results returned for the *Dominique Strauss-Kahn* query, and they included

heterogeneous sources. Therefore, this use case demonstrates the universality of our approach with respect to heterogeneous web data.

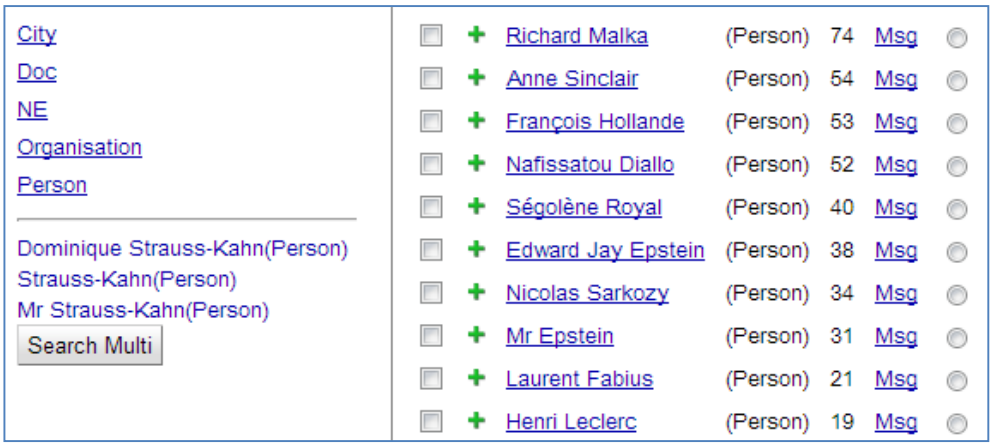


Figure 15: Related people in DSK case

In Figure 15 we can see the list of relevant people returned for *Dominique Strauss-Kahn*. We selected three aliases of the name (see the bottom left corner in Fig. 14 just above the *Search Multi* button) and the relevant people were returned, as you can see in the returned list. We did not initially know the type of relation, but each case can be separately explored by clicking on the *Msg* link in the respective row. Doing so returns the relevant documents for both entities. Richard Malaka (RM) is the lawyer of DSK. AS is DSK's wife. FH is the chairman of the Socialist Party (SP), of which DSK was a member. ND was the maid in the NY hotel where the scandal started. Originally, her name was not returned in the person list by our algorithm, because her first name (*Nafissatou)* was not in our gazetteer of valid first names. She was simply detected as a type-less named entity. By using the change type operation, we changed this entity to Person type. Other people such as SR, NS, or Mr. Epstein are also relevant.

LinkedIn Job Search

In this use case, we focused on crawling and searching LinkedIn job offers. We crawled more than 100,000 web pages, identified more than 70,000 job offers, and extracted more than 200,000 entities. The goal of the application was to match people's CVs with the job offers. We used strategies based on full-text and faceted search as well as those for graph search described in this thesis. Details of the application can be found in [DLUG12]. Information on the graph search part of the use case was described in [LAC11C].

We discovered that the full-text search approach supported by facets based on the extracted entities was more successful than graph search. We believe this was due to the following reasons:

- Objects of interest were usually documents (CVs or Job Offers) and not the entities mentioned in the text
- In the Skills category, extraction was not very successful.
- Locations were nodes with a high degree and they were usually starting points
- The assortativity coefficient of the network was positive

In order to improve the results, we modified the undirected graph to a directed one, where in most cases the edges were going in both directions. However, we have defined several types of restriction on the direction of the edges: *=>*Money*, *Doc=>JobTitle*, *Doc=>DocTitle*, *City=>JobLocation*, *City=>Location.*

These restrictions are defined for vertex types (*key* in vertex graph definition) and help differentiate among the edge types although, formally, our algorithm works with type-less edges. We have defined these rules for directed edges in order to infer job properties such as salary, job title, and job location. In the case of undirected graph, these properties were not correctly inferred in the majority of searches.

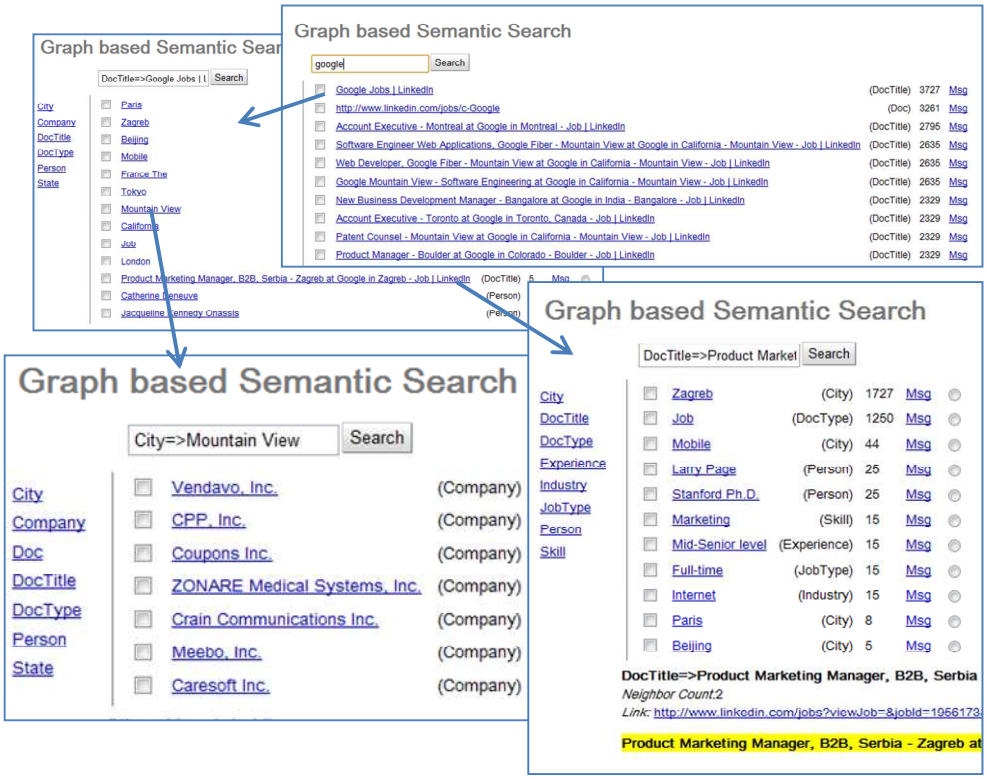


Figure 16: Graph Search in LinkedIn Job Offers

Figure 16, top right: full-text search in entities (graph nodes). When clicking on a link in the results, we started to navigate the entity graph and a list of relevant entities was returned. After clicking on *Google Jobs* (top right), the relevant entities such as locations, companies, or job offers were returned (top left). In the bottom left window, we see the list of relevant companies for the *Mountain View* location. In the bottom right window, we see the related entities for a specific job offer.

4.3 Gorilla Scandal

In the Gorilla Scandal[23] (Kauza Gorila in Slovak) use case we analyzed a leaked document from the Slovak Information Service. The document was one monolithic report. We have divided the report into smaller parts based on its paragraphs, and we have extracted human names, political parties, company names, dates, and amounts.

Relation discovery works quite well in some cases. The biggest problem is the Slovak morphology in which many word forms for the same entity exist. For example, the person for whom we provided an example in Figure 17 has 16 different morphological forms of the name. In the example, we selected only four of them. In order to search properly we would need to group these forms. We can merge them manually, but an automatic or semi-automatic approach would be needed in order to improve the search.

We noticed that the extracted network was assortative, similar to the LinkedIn network. We would like to further investigate if a search algorithm works better on disassortative networks only. This will be part of our future work.

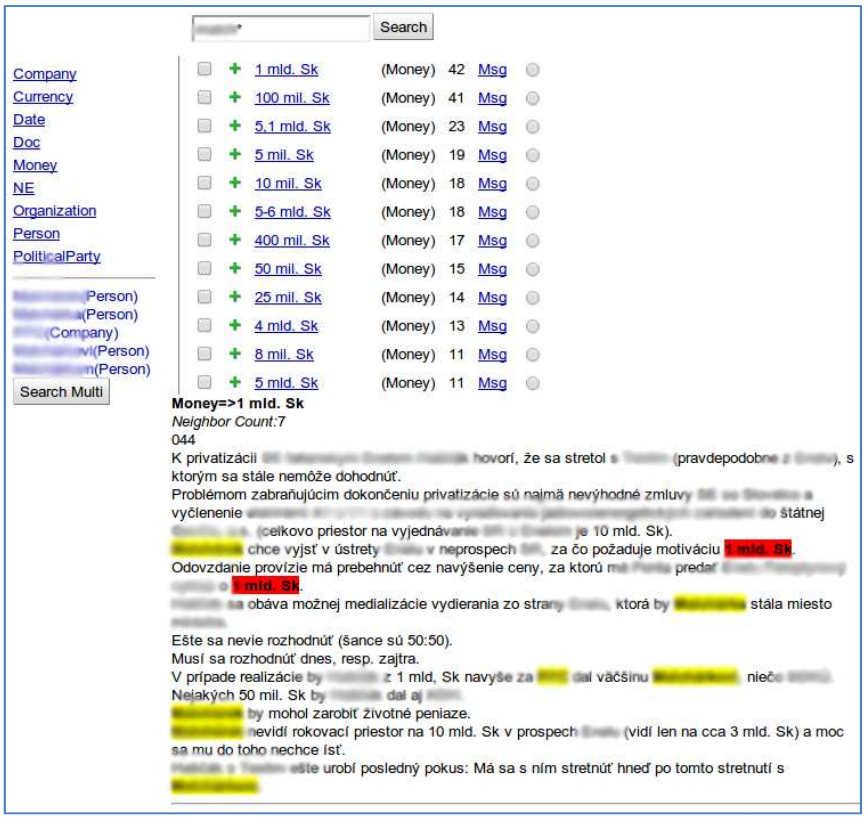


Figure 17: Search in Gorilla Scandal document

In Figure 17 you can see that we first did a full-text search for a specific person. In order to uncover all of the morphological forms, we put asterisks (*) at the end of the search. Then we selected four morphological forms of the same name. By clicking on

[23] http://en.wikipedia.org/wiki/Gorilla_scandal, http://sk.wikipedia.org/wiki/Kauza_Gorila

Company, we found the related companies for this person. We selected one of the related companies. Then by clicking on *Money* type, we discovered the list of the related money amounts for this person and the selected company. Finally, we selected one of the amounts. We clicked on the *Msg* link for the first amount and received the most relevant paragraph from among all of the related documents.

4.4 Entity relation discovery on non-text graphs

In this section we discuss two use cases in which we used methods for graph-based entity relation discovery on graphs constructed from sources other than text. We would like to show the possibility of future integration of structured and unstructured data in order to use the same approach for entity relation discovery.

Graph data form Agent-based Simulation

In this section we briefly describe the use and features of gSemSearch on the event data gathered from a Multi-agent simulation of interaction between angry civilians and soldiers. It was performed as part of EUSAS[24] project which focused on creating a tool for the simulation of civilians and training military personnel for operations in urban environment. The method is used for the data analysis of simulation runs which explore interesting events in simulation in order to analyze Measures of Effectiveness (MoE) or to discover potential problems in the simulation model. For example, the number of injuries and fatalities are important MoEs, and therefore it would be very useful to be able to discover the underlying causes leading to them in order to prevent such situations. Some information on this use case and the application of graph analysis was also published in [TAV12].

Figure 18: Graph Inference Tool GUI with Full-text search feature

The user interacts with the Graph Inference Tool (GIT/gSemSearch) by accessing the GIT URL via a web browser. The user enters the web application and can then proceed to search for an object he/she wants to analyze. The full text search is used for this feature (as seen in Figure 18). For example, if the user types *killed* and clicks the search button, the related value *killed* (MoE) is returned (Figure 18, left frame). If he/she searches for attack, all graph nodes (objects related to simulation) containing the string *attack* are returned.

[24] EUSAS - EDA project: European Urban Simulation for Asymmetric Scenarios (2010-2012) A-0938-RT-GC

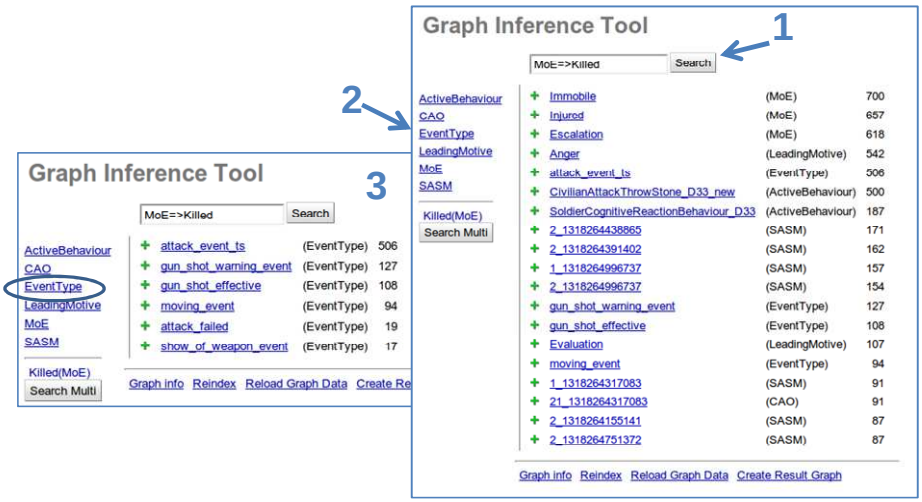


Figure 19: Exploring Killed (MoE) in GIT

When the user is interested in situations in which someone has been killed, he/she can add the *Killed* (MoE) to the list of objects used for later exploration by clicking on the green "+" icon in front of it. To select other related objects, the user clicks on the *Killed* object itself to explore its relations. GIT will then return all the related objects of various types for the *Killed* (MoE). As shown in Figure 19, the explored object will appear in the search box. The user can restrict the returned related objects by type so that only objects of Event types are returned (as seen on the left, Figure 19). As we can see from GIT analysis, dead civilians or soldiers are related to events like *attack_event_ts* (throwing of the stone by a civilian), *gun_shot_warning_event* (warning shot) or *gun_shot_effective* (effectively shot by soldiers). If we intend to explore an effective shot, we click on the plus icon next to it (Figure 20).

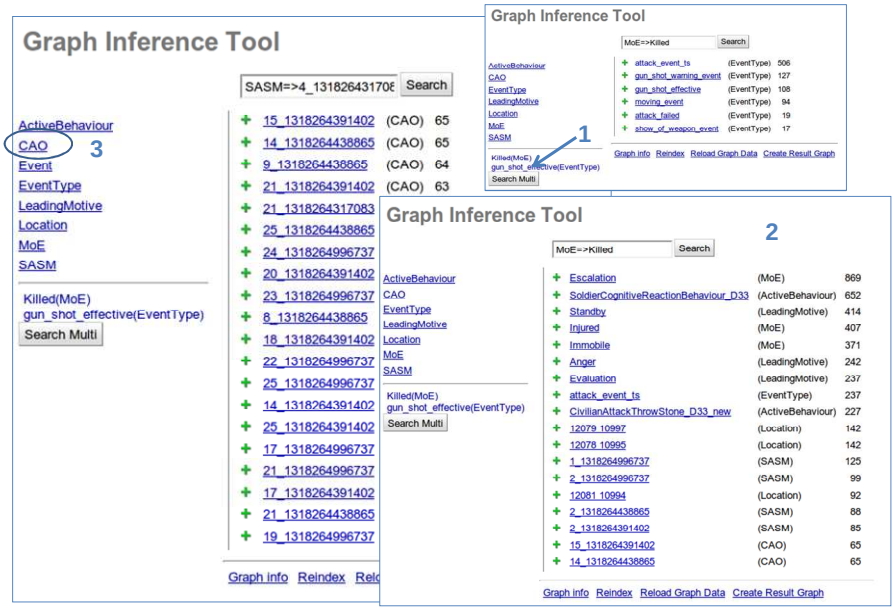


Figure 20: Exploring Killed (MoE) and gunshot event type

The user can continue the analysis by navigating through displayed objects. After we have selected a few objects (e.g., *Killed* (MoE) and *gun_shot_efective_event* event type), we can explore the related objects by clicking on the *Search Multi* button as seen in the top right frame of Figure 20. In the bottom right corner of same figure we see the mixed type list of the related entities returned for both selected objects (Figure 20, Screen 2). It can be restricted to show only the related civilians killed by a gun shot by clicking on *CAO* (Figure 20, Screen 3). We can further continue to specify and refine the conditions for search by selecting (clicking on "+" icon) a specific Civilian agent.

Finally, the user selects the intended objects to explore (event type, MoE, and Civilian), and by clicking on the *Search Multi* button the related results are returned (as seen in Figure 21). By clicking on *Create Results Graph*, one can generate a sub graph that can be visualized as a graph/network using the JUNG visualization library. The user can then interact with this graph, delete nodes, or zoom deeper into the graph. Support for actions like expanding and collapsing nodes for better visibility is planned. Since, unlike emails, we cannot explore relations by showing related text messages in event graphs, another type of visualization is needed. We have provided it by visualizing the graph of the related entities.

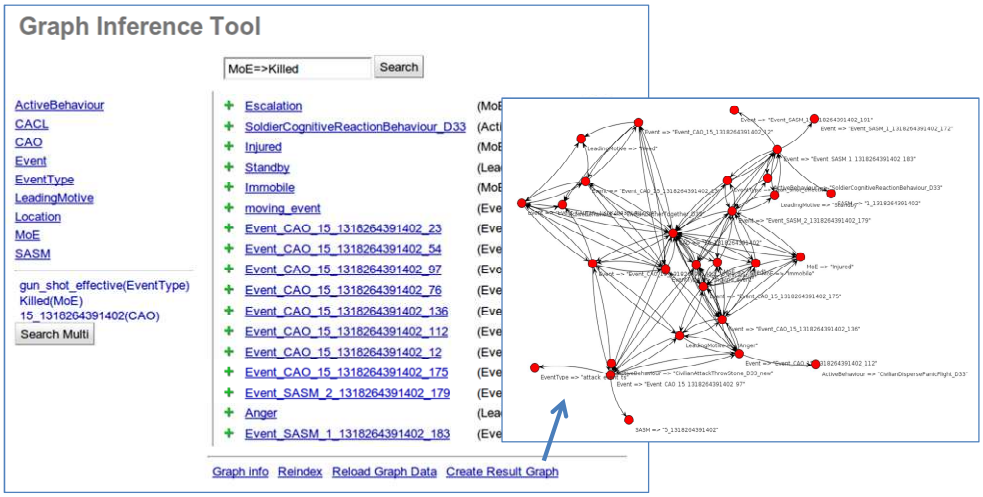


Figure 21: Explored results in GIT with Graph Visualization

The best way to explore and analyze the situation is to restrict the related results to Events only (as seen in Figure 22). The user can also visualize the Event graph which can be difficult to interpret (as you see in Figure 22, Screen 2), but when the user interacts with the graph by zooming, selecting, and deleting some of the nodes not needed for exploring the situation, the graph can become more readable (Figure 22, Screen 3). For example, here we see that one concrete civilian was shot by different soldiers multiple times (Figure 22, Screen 4). This example illustrates model calibration in which the civilian should be dead after two shots. This indicates that there is some bug in the model, the model implementation, or the model calibration. When the modeling expert uses the GIT tool, he/she can identify the problem by exploring the simulation logs. The web interface of GIT is used to identify and find interesting situations (usually related to MoEs), while the Graph Visualization is used

to explore the situation more deeply. For example, you can see 12 soldier events (Figure 22, on the left side) in the list related to different soldiers. Since these events are related to the *gunshot effective* event and a concrete civilian, we can assume that the effective gunshots with high relevance (10 out of 12 listed) were aimed at this particular civilian which is against the rules of engagement and should be investigated.

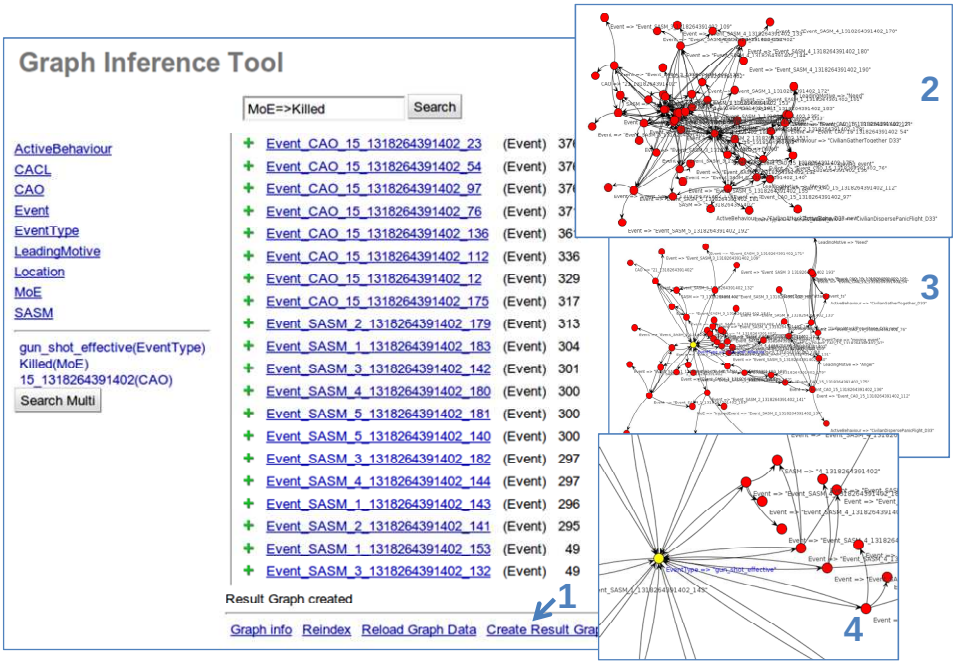


Figure 22: Exploring events related to a situation

LinkedData Simplified Graphs

We have conducted several experiments on entity relation search also on LinkedData graphs. Concretely we have focused on the DBLP (Computer Science Bibliography) dataset as well as the ACM datasets. Experiments showed that we can return relevant entity relations; however, it became evident that the simplified graph structure with type-less edges is not sufficient to explore the rich relations described by LinkedData. In order to use our approach on LinkedData graphs, we would need to redefine graph structure to include, explore, and use labeled graphs (edges with types/properties).

Since the DBLP dataset does not contain citations, we have focused more on ACM. The advantage is that through citations, one can get the related papers or authors for a paper or subject represented by the selected papers. The published ACM LinkedData database contains only publication older than 10 years.

As was already mentioned, we had to simplify the LinkedData RDF graph to be compatible with the graph structure of our semantic text graphs. More information on ACM graph simplification can be found in [MOJ12].

Figure 23: Search in ACM graph

In Figure 23 we see a full-text search for an entity (an author in this case). Two aliases for the same author are selected and a list of relevant items, publications, and other authors is returned. In this example, we see only the list of the author's publications and his co-authors, which is easy to infer from the graph. A more interesting example can be seen in Figure 24 in which, for two authors from the information retrieval field, the relevant entities are returned. The first item is *H.3.3*, which means *Information Search and Retrieval* in ACM classification[25], and the second item is a highly cited paper in the field about the first Google architecture. By clicking on any entity, we can explore the related entities.

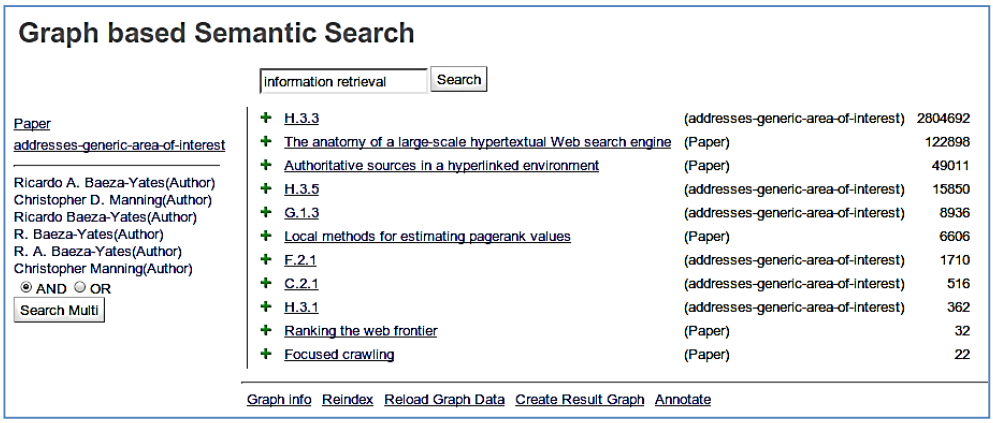


Figure 24: Relations in ACM Linked Data Graph

Through this experiment, we have demonstrated that our algorithm can also be used to some extent in finding relevant articles related to the selected authors, research fields, and articles. In the process of discovering relations, citation graph is exploited.

[25] http://www.acm.org/about/class/ccs98-html

5 CONCLUSION

In this thesis we have focused on entity relation discovery from unstructured text, where text was transformed into an information network with similar properties to other social or information networks. We have conducted experiments on several networks and graphs extracted from diverse text resources. We have shown and evaluated an interactive method of relation discovery available in the gSemSearch prototype.

We believe the information networks, such as the graph data presented in the thesis, can help to interconnect unstructured and structured data such as text documents (web pages, emails, documents) with the structured data (hyper text networks, social networks, LinkedData). When structured and unstructured data possess similar properties of small world information networks, we can apply common algorithms for search and exploration of entities and their relations. We believe this thesis has contributed in that direction.

References

[ANY05] ANYANWU, K. - MADUKO, A. – SHETH, A. (2005). SemRank: ranking complex relationship search results on the semantic web. In *Proceedings of the 14th international conference on World Wide Web* (WWW '05). ACM, New York, NY, USA, 117-127. DOI=10.1145/1060745.1060766

[AGA11] AGGARWAL, C. C. (Ed.). (2011). Social network data analytics. Springer. 502p, SBN 978-1-4419-8462-3

[BIRD06] BIRD, C. - GOURLEY, A. - DEVANBU, P. - GERTZ, M. – SWAMINATHAN A. (2006): Mining Email Social Networks. In: *MSR '06: Proceedings of the 2006 Workshop on Mining Software Repositories*. ACM, New York (2006) 137–143.

[CIG10] CIGLAN, M. - RIVIÈRE, É. – NØRVÅG, K. (2010): Learning to Find Interesting Connections in Wikipedia. In *Proceedings of the 2010 12th International Asia-Pacific Web Conference* (APWEB '10). IEEE Computer Society, Washington, DC, USA, 243-249. DOI=10.1109/APWeb.2010.62

[CIG10B] CIGLAN, M. – NØRVÅG, K. (2010): SGDB - Simple graph database optimized for activation spreading computation. *Proceedings of GDM'2010* (in conjunction with DASFAA'2010)

[CIG12] CIGLAN, M. – AVERBUCH, A. – HLUCHY, L. (2012): Benchmarking traversal operations over graph databases, Proceedings of GDM'12, IEEE ICDE Workshop, 2012

[CLA09] CLAUSET, A. - SHALIZI, C.R. – NEWMAN, M.E.J. (2009): Power-law distributions in empirical data. *SIAM Review* 51(4), 661-703 (2009). (arXiv:0706.1062, doi:10.1137/070710111)

[CRES97] CRESTANI, F. (1997): Application of Spreading Activation Techniques in InformationRetrieval. *Artif. Intell. Rev.* 11, 6 (December 1997), 453-482. DOI=10.1023/A:1006569829653

[DIN03] DINGLI, A. - CIRAVEGNA, F. – WILKS, Y. (2003): Automatic Semantic Annotation using Unsupervised Information Extraction and Integration. in *Proceedings of the K-CAP 2003* Workshop on Knowledge Markup and Semantic Annotation.

[DIX10] DIX, A. - KATIFORI, A. - LEPOURAS, G. - VASSILAKIS, C. - SHABIR, N. (2010). Spreading activation over ontology-based resources: from personal context to web scale reasoning. *International Journal of Semantic Computing*, 4(01), 59-102.

[DLUG12] DLUGOLINSKÝ, Š. - ŠELENG, M. - LACLAVÍK, M. - HLUCHÝ, L. (2012): Distributed Web-scale Infrastructure for Crawling, Indexing and Search with Semantic Support. In *Computer Science Journal*, Vol 13 No.4, pages 5-19, 2012, http://dx.doi.org/10.7494/csci.2012.13.4.5

[FAU12] FAUSCETTE, M. (2012): The Future of Email Is Social. White Paper; IBM IDC report; ftp://ftp.lotus.com/pub/lotusweb/232546_IDC_Future_of_Mail_is_Social.pdf

[CHAP05] CHAPANOND, A. - KRISHNAMOORTHY, M. S. - YENER, B. (2005). Graph theoretic and spectral analysis of Enron email data. *Computational & Mathematical Organization Theory*, 11(3), 265-281.

[CUNN06] CUNNINGHAM, H. (2006), Information Extraction, Automatic. In: Keith Brown, (Editorin-Chief) *Encyclopedia of Language & Linguistics*, Second Edition, volume 5, pp. 665-677. Oxford: Elsevier.

[CUNN11] CUNNINGHAM, H. ET AL. (2011): Text Processing with GATE (Version 6). University of Sheffield Department of Computer Science. 15 April 2011. ISBN 0956599311.

[FERR11] FERRUCCI, D. A. (2011). IBM's Watson/DeepQA. SIGARCH Comput. Archit. News 39, 3 (June 2011), -. DOI=10.1145/2024723.2019525 http://doi.acm.org/10.1145/2024723.2019525

[HASS12] HASSAN, A. - ABU-JBARA, A. - RADEV, D. (2012). Extracting Signed Social Networks From Text. TextGraphs-7, 6.

[JUD07] JUDGE, J. - SOGRIN, M. - TROUSSOV, A. (2007). Galaxy: IBM ontological network miner. In Proceedings of the 1st Conference on Social Semantic Web (Vol. 113).

[KLIMT04] KLIMT, B. – YANG, Y. (2004): Introducing the Enron Corpus. First Conference on Email and Anti-Spam (CEAS), 2004, http://www.ceas.cc/papers-2004/168.pdf, http://www.cs.cmu.edu/~enron/

[KVA10] KVASSAY M. - LACLAVÍK, M. - DLUGOLINSKÝ, Š. (2010). Reconstructing social networks from emails. In *DATESO 2010* : databases, text, specifications, objects. Eds J. Pokorný, V. Snášel, K. Richta. - Praha : MATFYZPRESS publishing house, 2010, p. 50-59. ISBN 978-80-7378-116-3.

[LAC09] LACLAVÍK, M. - ŠELENG, M. - CIGLAN, M. - HLUCHÝ, L. (2009). Ontea: Platform for pattern based automated semantic annotation. In *Computing and informatics*, 2009, vol. 28, no. 4, p. 555-579. (0.492 - IF2008). (2009 - Current Contents). ISSN 0232-0274.

[LAC10] LACLAVÍK, M. - KVASSAY M. - DLUGOLINSKÝ, Š. - HLUCHÝ, L (2010): Use of Email Social Networks for Enterprise Benefit. In: *IWCSN 2010*, IEEE/WIC/ACM WI-IAT, 2010, pp 67-70, DOI 10.1109/WI-IAT.2010.126

[LAC11] LACLAVÍK, M. - DLUGOLINSKÝ, Š. - ŠELENG, M. - KVASSAY M. – GATIAL, E. – BALOGH, Z. - HLUCHÝ, L (2011): Email Analysis and Information Extraction for Enterprise Benefit. In *Computing and Informatics*, 2011, vol. 30, no. 1, p. 57-87. ISSN 1335-9150, Special Issue on Business Collaboration Support for micro, small, and medium-sized Enterprises

[LAC11B] LACLAVÍK, M. - DLUGOLINSKÝ, Š. - KVASSAY M. - HLUCHÝ, L (2011): Email Social Network Extraction and Search. In NextMail 2011 workshop, WI-IAT 2011, In *The 2011 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*. IEEE Computer Society, 2011, p. 373-376. ISBN 978-0-7695-4513-4

[LAC11C] LACLAVÍK, M. - ŠELENG, M. - CIGLAN, M. - DLUGOLINSKÝ, Š. - HLUCHÝ, L. (2011). gSemSearch: Objavovanie relácií v kolekciách textových a grafových dát. In 6th Workshop on Intelligent and Knowledge Oriented

Technologies : *WIKT 2011 proceedings*. - Košice : Centre for Information Technologies, FEI TU Košice, 2011, p. 1-5. ISBN 978-80-89284-99-3.

[LAC12] LACLAVÍK, M. - DLUGOLINSKÝ, Š. - ŠELENG, M. - CIGLAN, M. - HLUCHÝ, L. (2012). Emails as graph: relation discovery in email archive. In Proceedings of the 21st international conference companion on World Wide Web (*WWW '12 Companion*). ACM, New York, NY, USA, 841-846, http://www2012.wwwconference.org/proceedings/companion/p841, DOI=10.1145/2187980.2188210.

[LAC12B] LACLAVÍK, M. - DLUGOLINSKÝ, Š. - ŠELENG, M. - CIGLAN, M. – TOMAŠEK, M. - KVASSAY, M. - HLUCHÝ, L. (2012). Lightweight semantic approach for enterprise search and interoperability. In *CEUR Workshop Proceedings*: InteropVlab.IT 2012. - CEUR, 2012, p. 35-42. ISSN 1613-0073.

[LAC12C] LACLAVÍK, M. (2012): Improving entity and relation discovery by user interaction with semantic graphs. In 7th *Workshop on Intelligent and Knowledge Oriented Technologie*s: P. 161-164. - Bratislava: Nakladateľstvo STU, 2012. ISBN 978-80-227-3812-5.

[LARKC11] GRINBERG, M. - STEFANOV, H. - STEFANOV, K. – PEIKOV, I. (2012): D2.4.3 Spreading activation components v3. LarKC Project Deliverable, http://www.larkc.eu/wp-content/uploads/2008/01/LarKC-D2.4.3-Spreading-activation-components-v3.pdf

[LUM07] LUMSDAINE, A. - GREGOR, D. - HENDRICKSON, B. – BERRY, J. (2007). Challenges in Parallel Graph Processing. Parallel Processing Letters, 17(1):5-20, March 2007.

[MAL09] MALEWICZ, G. - AUSTERN, M. H. - BIK, A. J.C. - DEHNERT, J. C. - HORN, I. - LEISER, N. – CZAJKOWSKI, G. (2009). Pregel: a system for large-scale graph processing - "ABSTRACT". In PODC '09. ACM, New York, NY, USA, 6-6, 2009, DOI=10.1145/1582716.1582723

[MIH11] MIHALCEA, R. – RADEV, D. (2011) *Graph-based Natural Language Processing and Information Retrieval*, Cambridge University Press, ISBN: 9780521896139, 208 pages

[MIL08] MILNE, D. - WITTEN, I.H. (2008) Learning to link with Wikipedia. In *Proceedings of the ACM Conference on Information and Knowledge Management* (CIKM'2008), Napa Valley, California.

[MIN08] MINKOV, E. (2008). Adaptive graph walk based similarity measures in entity-relation graphs (Doctoral dissertation, University of Texas at Austin).

[MIN12] MINKOV, E. - COHEN, W. W. (2012). Graph Based Similarity Measures for Synonym Extraction from Parsed Text. TextGraphs-7, 20.

[MOJ12] MOJŽIŠ, J. - LACLAVÍK, M. (2012): Navigácia v zjednodušenom LinkedData grafe. In 7th *Workshop on Intelligent and Knowledge Oriented Technologies*. - Bratislava : Nakladateľstvo STU, 2012, p. 15-18. ISBN 978-80-227-3812-5

[NEW03] NEWMAN, M. E. J. (2003). Mixing patterns in networks. *Physical Review* E 67 (2): 026126.

[NUTT04] NUUTILA, E., - TÖRMÄ, S. (2004). Text Graphs: Accurate Concept Mapping with Well-Defined Meaning. In Proceedings of the First International Conference on Concept Mapping.

[PAN06] PANTEL, P. – PENNACCHIOTTI, M. (2006). Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics* (ACL-44). Association for Computational Linguistics, Stroudsburg, PA, USA, 113-120. DOI=10.3115/1220175.1220190

[SAL88] SALTON, G. – BUCKLEY, C. (1988): On the Use of Spreading Activation Methods in Automatic Information Retrieval. Technical Report. Cornell University, Ithaca, NY, USA.

[SCH08] SCHUMACHER, K. - SINTEK, M. – SAUERMANN, L. (2008): Combining fact and document retrieval with spreading activation for semantic desktop search. In *Proceedings of the 5th European semantic web conference on The semantic web: research and applications* (ESWC'08), Springer-Verlag, Berlin, Heidelberg, 569-583.

[SUCH07] SUCHAL J. (2007): On Finding Power Method in Spreading Activation Search. In: *SOFSEM 2008: Volume II – Student Research Forum*, p. 124-130.

[SUCH10] SUCHAL, J. – NAVRAT, P. (2010): Full Text Search Engine as Scalable k-Nearest Neighbor Recommendation System. In: *Artificial Intelligence in Theory and Practice III IFIP Advances in Information and Communication Technology*, 2010, Volume 331/2010, 165-173.

[STR01] STROGATZ S. H. (2001): Exploring complex networks, Nature 410, 268-276, doi:10.1038/35065725

[TAV12] TAVCAR, A. - GAMS, M. - KVASSAY, M. - LACLAVIK, M. - HLUCHY, L. - SCHNEIDER, B. - BRACKER, H. (2012): Graph-based analysis of data from human behaviour simulations, *Applied Machine Intelligence and Informatics (SAMI)*, 2012 IEEE 10th International Symposium on , vol., no., pp.421-426, 26-28 Jan. 2012, doi: 10.1109/SAMI.2012.6209003

[TROUS09] TROUSSOV, A., PARRA, D., & BRUSILOVSKY, P. (2009). Spreading Activation Approach to Tag-aware Recommenders: Modeling Similarity on Multidimensional Networks. In: D. Jannach, et al. (eds.) *Proceedings of Workshop on Recommender Systems and the Social Web* at the 2009 ACM conference on Recommender systems, RecSys '09, New York, NY, October 25, 2009.

[ULAN12] ULANOFF L. (2012). Google Knowledge Graph Could Change Search Forever. http://mashable.com/2012/02/13/google-knowledge-graph-change-search/

[ZH05] ZHANG,M. - SU, J. - WANG, D. - ZHOU, G. - TAN C. L. (2005): Discovering relations between named entities from a large raw corpus using tree similarity-based clustering. In Proceedings of the Second international joint conference on Natural Language Processing (IJCNLP'05), Springer-Verlag, Berlin, Heidelberg, 378-389. DOI=10.1007/11562214_34

6 APPENDICES

6.1 Emails as Graph: Relation Discovery in Email Archive

[LAC12] LACLAVÍK, Michal - DLUGOLINSKÝ, Štefan - ŠELENG, Martin - CIGLAN, Marek - HLUCHÝ, Ladislav. Emails as graph: relation discovery in email archive. In WWW´12 Companion Proceedings of the 21st International Conference companion on World Wide Web. - New York : ACM, 2012, 841-846. ISBN 978-1-4503-1230-1.

Emails as Graph: Relation Discovery in Email Archive

Michal Laclavík
Institute of Informatics,
Slovak Academy of Sciences
Dúbravská cesta 9, Bratislava
+421-2-59411256
laclavik.ui@savba.sk

Štefan Dlugolinský
Institute of Informatics,
Slovak Academy of Sciences
Dúbravská cesta 9, Bratislava
+421-2-59411207
stefan.dlugolinsky@savba.sk

Martin Šeleng
Institute of Informatics,
Slovak Academy of Sciences
Dúbravská cesta 9, Bratislava
+421-2-59411256
martin.seleng@savba.sk

Marek Ciglan
Institute of Informatics,
Slovak Academy of Sciences
Dúbravská cesta 9, Bratislava
+421-2-59411176
marek.ciglan@savba.sk

Ladislav Hluchý
Institute of Informatics,
Slovak Academy of Sciences
Dúbravská cesta 9, Bratislava
+421-2-54771004
hluchy.ui@savba.sk

ABSTRACT
In this paper, we present an approach for representing an email archive in the form of a network, capturing the communication among users and relations among the entities extracted from the textual part of the email messages. We showcase the method on the Enron email corpus, from which we extract various entities and a social network. The extracted named entities (NE), such as people, email addresses and telephone numbers, are organized in a graph along with the emails in which they were found. The edges in the graph indicate relations between NEs and represent a co-occurrence in the same email part, paragraph, sentence or a composite NE. We study mathematical properties of the graphs so created and describe our hands-on experience with the processing of such structures. Enron Graph corpus contains a few million nodes and is large enough for experimenting with various graph-querying techniques, e.g. graph traversal or spread of activation. Due to its size, the exploitation of traditional graph processing libraries might be problematic as they keep the whole structure in the memory. We describe our experience with the management of such data and with the relation discovery among the extracted entities. The described experience might be valuable for practitioners and highlights several research challenges.

Categories and Subject Descriptors
H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval. H.4 [**Information Systems Applications**]: H.4.3 Communications Applications: Electronic mail;

General Terms
Algorithms, Measurement, Performance, Design, Experimentation

Keywords
email, search, relation discovery, Enron corpus, graph data, social networks.

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

1. INTRODUCTION
Graphs or networks appear often as a natural form of data representation in many applications: *Social Networks* (contain high amount of graph data like friend networks, information about the interaction among other artefacts like statuses, messages, photos or tags), *Call networks* (network of communicating people including audio, video or SMS communication with additional data such as location), *Internet* (web graph of interconnected web pages), *Wikipedia* (network of Wikipedia concepts including hierarchies, themes or language variations), *LinkedData*[1] (fast growing semantic network data containing metadata about people, geo-locations, publications and other entities), *Emails* (social networks are included also in email communication [2], which can be connected to other objects mentioned in emails like contact information, people, organizations, documents, links, or time information).

Analysis of email communication allows the extraction of social networks with links to people, organizations, locations, topics or time information. Social Networks included in email archives are becoming increasingly valuable assets in organizations, enterprises and communities, though to date they have been little explored. We believe that email communication and its links to other organizational as well as public resources (e.g. LinkedData) can be valuable source of information and knowledge for knowledge management, business intelligence or better enterprise and personal email search. The future of email [18] is in interconnecting email with other resources, services (like social networks or collaboration tools), data and entities, which are present in email. Our work tries to make this integration possible. In this paper we discuss email communication and email archive as a graph or network structure. We describe extracted graph data as a ready to use data source for experimentation with information networks. We illustrate the process of the entity network extraction on the well-known Enron email corpus[2][1]. Enron corpus was analyzed in many ways including social

[1] http://linkeddata.org/

[2] http://www.cs.cmu.edu/~enron/

(communication) networks [18] and its visualization[3]. We have created Enron Email Graph, which contains various entities and social networks extracted from emails. Each processed email has its own node in the graph with connections to named entities (NE) extracted from this email such as people, email addresses, telephone numbers. Named entities of the same type and value (e.g. "Person" => "John") are unique in the graph, so one entity found in different emails is presented only once in the graph, but it is connected to all the emails from which it has been extracted. This is the simplest approach possible to deal with the synonymy and polysemy of the extracted NE. We show the strength of this approach for retrieving related entities when given broader context. Edges in the graph are links between NEs representing co-occurrence in the same email part, paragraph, sentence or a composite NE. The Enron Email Graph exhibits the "small world" property typical for many information networks, and can be used for further experimenting. In the paper, we describe the entity graph creation process and study its properties.

As the size of the real graph data grows, there must be an adequate development in the field of graph data management. Typically, software libraries designed for graph data processing store the whole structure in the memory. This is a serious drawback when the size of the data exceeds the available memory. Recently, there has been a growing interest in managing graph data persistently. Several important research and development directions include: triple stores (semantic web databases focused on storing semantics in the form of triples like Virtuoso, Sesame, OWLim or SHARD[4]), Graph databases or graph APIs (Neo4j, Virtuoso, SGDB[5], or JUNG, which allow graph manipulation, traversing or persistent data storage), Blueprints[6] (a common Java API for graph databases, similarly as JDBC for relational databases). Fast graph traversing is the most important feature when querying large graphs. The challenge is to make the graph querying scalable, since graph traversing has to deal with random access pattern to the nodes [15]. Due to this fact, graph databases try to load most of the data into memory. Scalable processing on parallel, shared-nothing architectures is just emerging, since even big enterprises like Facebook or Google still need to solve large and scalable graph processing. Google published its Pregel [3] solution for graph batch processing. Similarly, open-source solutions like Hama or Giraph[7] based on Pregel idea are emerging. However, to the best of our knowledge, there is no scalable solution yet for real-time graph querying. The work presented in this paper goes in this direction and builds on our previous work [4, 5, 6] in the email archive processing, information extraction (IE), email social network extraction and relation discovery.

In our previous work [6], we have described similar network structure extracted from the Enron corpus, where we have experimented with the spread of activation algorithm but this only worked on sub-corpuses covering 1-10% of the entire Enron corpus. In this paper we describe the approach working on the entire Enron Corpus, where we had to change the algorithm due to to poor performance. After processing the entire corpus, we are able to compute and discuss the network properties of the extracted information network. We discuss advances in relation discovery, graph processing infrastructure and querying user interfaces with focus on relation discovery in email communication. We have also added multi node relation discovery and improved the entire relation search interface relative to our previous work [6].

This paper is structured as follows: section 2 discusses the approach for entity extraction and graph/network creation. It also presents the statistics and properties of the extracted network. Section 3 discusses the relation discovery approach. First, we discuss the problem of fast graph traversing algorithms in large graph structures and in real-time graph querying. Next, entity relation discovery algorithm based on spread of activation is discussed, followed by a description of the user interface for relation search and performance evaluation similar to [6], but working on the entire Enron corpus.

2. GRAPH CORPUS EXTRACTION

In this chapter we describe the entity extraction part of the Enron Graph Corpus creation process. The Enron Graph Corpus is built up from the Enron Email Corpus [1]. At first, we focus on the information extraction (IE) as well as the tree and graph construction from emails. Then we explain the processing of the whole corpus on a Hadoop[8] cluster. Furthermore, we provide information on the extracted Enron Graph Corpus and finally we discuss the properties of the extracted information network.

2.1 Entity, Tree and Graph Extraction from Email Corpus

In our previous work [4, 5, 6] we describe the extraction of email social networks. In order to reveal the social network graph hidden in the email communication, the important task is to identify objects and their properties in emails. For object identification we use Ontea IE techniques [5] based on regular expressions and gazetteers as can be seen in Figure 1. Applied patterns and gazetteers extract key-value pairs (object type – object value) from email textual content as it is displayed in the middle of Figure 1. If there is textual data present in binary form (e.g. PDF attachment) it is, if possible, converted to text before the information extraction process. Ontea is able to detect message replies inside emails. In the presented network we ignore the entities detected in the replies, but it would make sense to experiment with the entities in replies as well. The extracted key-value pairs are then used to build the tree (right side in the Figure 1) and the graph of social network (Figure 2). So the social network contains not only the communicating parties but also the related extracted entities, which can be further explored.

The Ontea IE tool is able to connect other extraction/annotation tools like GATE[9] or Stanford CoreNLP[10]. In our experiment we have also connected WM Wikifier[11] to link email communication with external public knowledge base; however at the end we did not include this information to the graph corpus. The reason was imprecision on the subset of example emails. E.g., Wikifier has

[3] http://hci.stanford.edu/jheer/projects/enron/

[4] http://sourceforge.net/projects/shard-3store/

[5] http://ups.savba.sk/~marek/sgdb.html

[6] https://github.com/tinkerpop/blueprints/wiki/

[7] http://incubator.apache.org/giraph/

[8] http://hadoop.apache.org/

[9] http://gate.ac.uk/

[10] http://nlp.stanford.edu/software/corenlp.shtml#About

[11] http://www.nzdl.org/wikification/

annotated *front desk* text as *Receptionist*[12], detected *Executive Director*[13] and also recognized text *north tower* as *List of tenants in One World Trade Center*[14]. On a larger test set, there have been a lot of false annotations like songs, albums or annotated abbreviations like *CC*, *DSL*, *ASAP*. This kind of annotation is not very useful, but we believe that interconnecting public LinkedData with email content can have benefit in email exploration, relation search or classification.

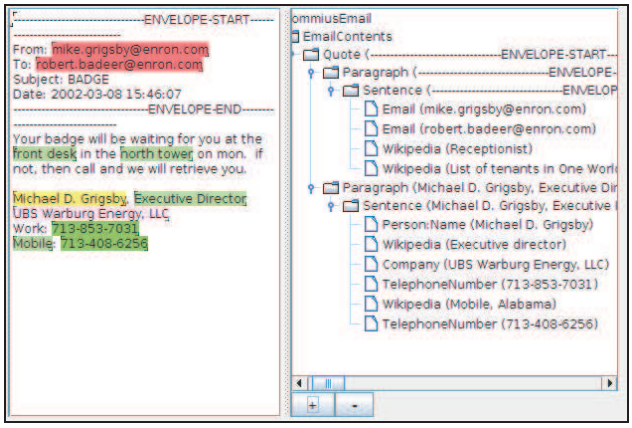


Figure 1. User interface of the IE tool Ontea [5] with highlighted extracted objects (left) and tree structure (right), which is used to build social network graph (Figure 2).

We can see a graph built from two emails in Figure 2. Note the two telephone numbers, company name and person name nodes connected to two different sentence nodes in Figure 2. These entities have been found in both emails, but they are presented only once in the graph (they are unique). For both nodes and edges we know also the numeric value of node or edge occurrence in the collection. This can be used as edge or node weight. So far we did not use it in relation discovery algorithm.

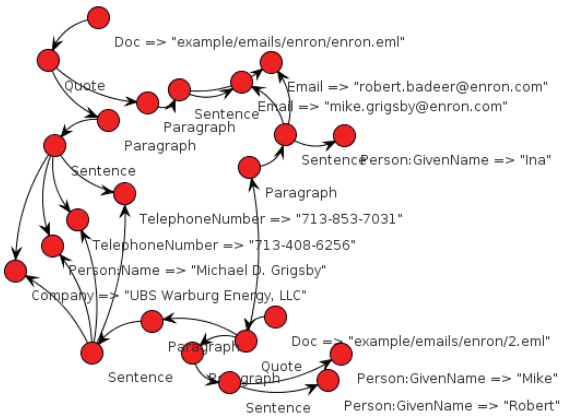


Figure 2. Social Network Graph built from two Enron emails

Processing Enron Corpus on Hadoop. We have wrapped Ontea functionality into Hadoop MapReduce library similarly as we did in our previous MapReduce experiment [7], since processing on single machine was time consuming and took several hours. It takes now about 90 minutes to process the whole Enron Email Corpus on our testing 8-node Hadoop cluster (Intel® Core™ 2 CPU 2.40GHz with 2GB RAM hardware on all machines). We have used a different version of Enron corpus in our previous work [7]. Now we use Enron Email corpus [1], which follows the structure of user mailboxes, where each email is a single file on the disk. We have created HDFS continuous file from this archive to process it much faster on a Hadoop Cluster.

2.2 Enron Graph Corpus

Here, we describe properties of the Enron Email Graph corpus.

Extracted entities. The resulting graph contained 8.3 millions vertices and 20 million edges extracted from 0.5 million messages. The graph comprised the following number of nodes with identified type: addresses (4,997 instances), CityName (1,550 instances), Company (52,286), DateTime (228,175), Email (162,754), MoneyAmount (28,992), Paragraph (2,631,292), Person (167,613), Quote (533,007), Sentence (3,800,504), Telephone Number (26,013) and WebAddress (105,610). Such representation allows the relation discovery as presented in the next section.

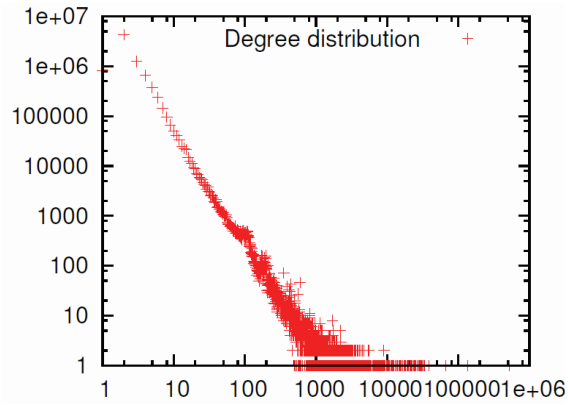


Figure 3. Enron Graph node degree distribution

Graph properties. Extracted graph has properties of small world information networks similar to Wikipedia or web graph. In Figure 3 we can see the distribution of the node degree with power law distribution coefficient 1.9 computed according to [8]. From other properties, Assortativity coefficient [17] was with negative value -0.01942, denoting that the network is disassortative, similarly to other information networks. Average local clustering coefficient of the network is high: 0.292 meaning that 29% of nodes in a graph tend to cluster together. Average shortest path on sample of graph data is 6.58 hops.

3. RELATION DISCOVERY

Presented Enron Graph Corpus can be used for graph querying experiments. In our previous work [6], we have used only a fraction of this corpus to discover relation among entities. We have developed Email Social Network prototype, which was discussed in [4, 5, 6]. In this paper we discuss new advancements in scalability using SGDB graph database and user interaction using gSemSearch tool.

[12] http://en.wikipedia.org/wiki/Receptionist

[13] http://en.wikipedia.org/wiki/Executive_Director

[14] http://en.wikipedia.org/wiki/List_of_tenants_in_One_World_Trade_Center

3.1 Real-time querying and spread of activation

In our approach we use spread of activation on the graph of multidimensional social network in a similar way as IBM Galaxy [9], where a concept of multi-dimensional social network for text processing was introduced. Spread of activation is also used on the Slovak website Foaf.sk [12, 13] for discovering relations between people and enterprises in Slovak business register, or in recommendation systems [10, 13] as well as in relation discovery in Wikipedia [14].

As we mentioned in the Introduction section, random node access is the key problem for fast graph traversing [15], which is also used in spread of activation algorithm. Simple Graph Database SGDB [11] was developed to be optimized for spread of activation. SGDB stores information about nodes and edges in an optimized form of key-value pairs.

In our previous implementation [6] we have used in-memory graph with JUNG graph library, where we could not even load the full Enron Graph corpus presented in this paper. Currently we use SGDB on a single machine and achieve satisfactory results (Section 3.3 discusses the performance evaluation) on the whole Enron Graph Corpus.

According to the best of our knowledge, SGDB [11] is the best graph engine for real-time graph querying [16]. In our future work we would like to go further, create scalable graph querying solution on a shared-nothing architecture cluster. One idea how to scale it is to use a distributed key-value store instead of a single machine key-value store. However, making it scalable will need to involve other techniques, and difficulties in communication and caching can arise.

When performing a spread of activation, we traverse only a part of the whole network, but this part grows quite fast with the depth of search, because we deal with small world networks, which have short paths between the nodes. After a few levels of activation, classical spread of activation algorithm can reach the whole graph. Therefore we still need to optimize the spread of activation algorithm (or other relation discovery algorithm) even when fast traversing infrastructure like SGDB is used. Most of the algorithms use modifications of Breadth First search and thus the depth of search needs to be optimized for each query. We have experimentally discovered that we cannot set up a common level of depth for different node relations discovery in information networks such as Enron Graph Corpus to achieve both: satisfactory relevant result and satisfactory performance, because the graph topology is different in each case. It is important to deal somehow with the high-degree nodes .

In our current implementation (algorithm in the next column) we use a simple approach of Breadth First traversing, which is limited to visit only *n* nodes. The algorithm skips nodes with higher degree (higher number of neighbor nodes) than the number of remaining nodes to be visited. When a node is skipped, we process the next node in the queue. We have experimentally set *n* number to 10,000 nodes to have a reasonable search time (around one second) and satisfactory relevant results.

We are using *LinkedList* as queue for nodes to be processed. For each node we simply ask for the number of its neighboring nodes, calculate the activation value accordingly and decide if we can explore the node or not. In *count* variable we hold the number of the nodes to be visited, which is decreased by the number of the processed nodes. In the future we should also consider the number of the edges between the nodes, or the strength of their "bond". At present, if there is more than one edge between two nodes, it has no effect on the activation.

```
private void computeRelatedBreadthFirst(Result start) {
    LinkedList<Result> rLL = new LinkedList<Result>();
    rLL.addLast(start);
    int count = visitNodeCount;
    rM.put(start, (double) count);
    vNodes++;
    while (!rLL.isEmpty() && count >= 0) {
        Result r = rLL.removeFirst();
        visited.add(r);
        int nCount =  g.g.getNeighborCount(r);
        double v = rM.get(r)/(double)nCount;

        //if value is to low we do not activate more
        if (v < threshold)
            continue;
        if (nCount<=count) {
            Collection<Result> rC = g.g.getNeighbors(r);
            for (Result result : rC) {
                if (!visited.contains(result)) {
                    rLL.addLast(result);
                }
                visited.add(result);
                double val = v;
                if (rM.containsKey(result))
                    val += rM.get(result);
                rM.put(result, val);

                vNodes++;
            }
            count -= nCount;
        }
    }
}
```

The algorithm ends up in reasonable times (around one second – based on the setting for the number of visited nodes) and still returns satisfactory relation results, but it can also fail if we want to compute relation for the nodes with high degree. For example, if we would search for relations to town *Hudson* or state *Texas*. Such entities have too many connections in Enron Graph Corpus. It does not make sense to infer entities related to *Texas* but it can make sense to infer entities related to a concrete person as well as *Texas* at the same time. In our current approach, *Texas* would be just ignored. In our future work we would like to consider the results on the path that activated *Texas* from other starting nodes as the relevant results.

3.2 gSemSearch

In our previous work [5, 6] we have started with creating Email Social Network Search user interface, which was extended for current Graph Semantic based Search (gSemSearch) tool. To compare advances with status reported in [6], gSemSearch now supports multiple node selection and activation. Moreover, it supports multiple node highlighting and the whole interface was improved. We had redesigned it to work with Blueprints[15] graph API. This way users are able to test and experiment on Enron Graph Corpus with any Blueprints compatible graph manipulation and storage framework, thus it can connect also to SGDB, which is Blueprints compatible.

To sum up, the gSemSearch functionality and its user interface allow relation discovery, where a user can perform a full-text search (e.g., *Gr***by* surname in Figure 4), then select starting nodes (e.g., two variations of person names of *Michael Gr***by* and *UBS* company in Figure 4 on the left) and search for the related nodes. A list of nodes with mixed type is returned. It can

[15] https://github.com/tinkerpop/blueprints/wiki/

be restricted to one node type by clicking on selected type (e.g. *TelephoneNumber* in Figure 4 on the left). This will return nodes of the desired type as can be seen in Figure 4 (related phone numbers). Prototype suggests that starting nodes and return results are related but does not suggest the type of relation. The reason and the type of the relation can be discovered by clicking on the *Msg* links next to the result nodes in the list. This will highlight starting nodes in the most related email message by yellow and selected node by red color (note that same objects can be present in multiple messages).

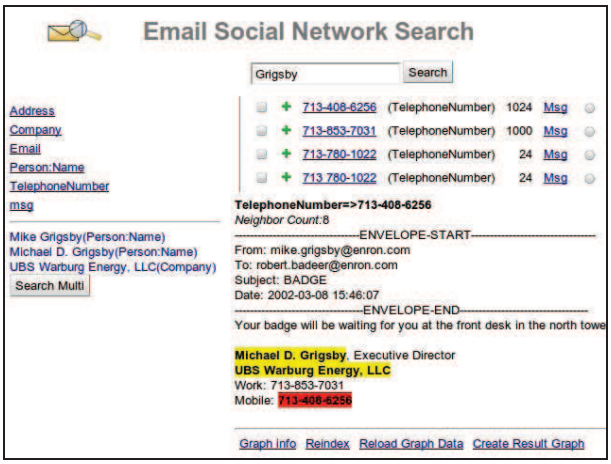


Figure 4. gSemSearch user interface.

In addition the gSemSearch user interface supports actions like nodes merging, deleting or changing the node type.

We also use a unique approach for synonymy and polysemy of the explored entities (ambiguity and disambiguation). If an entity is represented by more than one node (multiple aliases, similarly as the person in Figure 4), we can use two methods to explore the entities related to such an entity. We can either select all the aliases and search for the nodes related to this node cluster, or we can merge the aliases to a single node and explore its relations as if it were a single node. Both approaches have some drawbacks, but by their combination satisfactory results can be achieved. Another problem is if the same string represents two different entities. We do not provide automatic disambiguation during the extraction, so two different people with same name will be presented as one node in the graph. However, if some extra auxiliary information is known about the nodes, for example an address or company related to the person, the person node can be selected along with these related nodes, and then the search can be performed for other entities related to this multiple selection. This way the sub-graphs related to the other person represented by the same named entity either will not be explored at all, or will be explored/activated only partially.

We have also tested the gSemSearch relation discovery on other data types like graphs extracted from BBC news, LinkedIn job offers and event graphs of agent based simulations, so we see the possibility to explore our relation discovery approach and user interface in other domains, where data can be represented by graph/network structures with properties of information networks.

3.3 Performance Evaluation

In [4] we have evaluated success (precision and recall) of the IE and the success rate of relation discovery (the spreading activation algorithm) with satisfactory results [4]. Precision of discovered relation was 60% on Spanish email dataset and 77% on English one. But most of errors were introduces by imperfect information extraction. When ignoring information extraction errors, precision of relation discovery was about 85% [4]. In current implementation we did not evaluated precision of discovered relations, but it should be about similar, since we have tested relation discovery of updated algorithm on same test cases. The prototype and the algorithm were further refined with the focus on higher precision of IE results (impacting also results of relation discovery). Performance scalability was tested in [6], where satisfactory results were not achieved, but we have highlighted several possibilities for the improvement. We have implemented them and in Table 1 we present the performance evaluation similar to the evaluation in [6] but on the full Enron Graph Corpus. While in [6] we tested with 50,000 messages and less than 1 million of nodes, now the algorithm and infrastructure scales well on 8 million of nodes and 500,000 messages.

Table 1. Search time evaluation on Enron Graph Dataset for chosen entities (nodes)

Person:Name=>Mike Gri*by**	
Search Response Time (ms)	1, 195
Visited	9,441
Visited Unique	4 ,423
TelephoneNumber=>713 780-12**	
Search Response Time (ms)	378
Visited	4, 092
Visited Unique	2 ,627
Address=>6201 M*ow Lake, Houston, TX 77057**	
Search Response Time (ms)	171
Visited	6 ,188
Visited Unique	4, 078
Email=>ina.ra*l@enron.com**	
Search Response Time (ms)	615
Visited	7, 052
Visited Unique	3, 836

We have performed the same searches as in [6] on full Enron Graph Database for 4 different types of objects: person, telephone number, address and email address as seen in Table 1. The selected different types of entities represent different topology of related sub-graphs explored during graph traversal. For example an email address is usually connected to many nodes directly, while telephone number or address is connected to just a few sentence nodes. When searching for related nodes, different depth of graph traversing needs to be explored for different object types. We achieved this by using algorithm presented in this paper. The response time was computed as the average from 5 searches. As we have mentioned earlier, the algorithm visits only n nodes while traversing the graph, where n was set experimentally to 10,000. Thus we see that the number of visited nodes is less than 10,000 and the number of unique visited nodes is even smaller. The time of search is usually lower than 1 second, but it varies (from 171 ms to 1,195 ms in our experiment) based on the cached data of the underlying key-value store infrastructure.

4. CONCLUSION AND PERSPECTIVE

In this paper we have provided information about the whole Enron email corpus as a graph data resource for graph query experimentation, which is available online[16]. In addition we describe our tool gSemSearch[17] that allows users to experiment with relation discovery over the network extracted from email archives. We would like to encourage researchers to work with the presented corpus and query user interface, which can boost research in the area of large graph querying.

We believe the information networks, such as the graph data presented in this paper, can help to interconnect email with the enterprise or community data as well as LinkedData or other public data sources. This would allow using email archives as a knowledge base or (in enterprise contexts) exploiting them for business analytics.

In our future work we plan to interconnect email graph data with other resources, monitor events, activities or tasks within enterprise or in cross enterprise context in order to provide searchable knowledge base as analytical tool or tool helping in collaboration and interoperability.

5. ACKNOWLEDGMENTS

I would like to thank Marcel Kvassay for proofreading the paper. This work is supported by projects VENIS[18] FP7-284984, TRA-DICE APVV-0208-10, SMART II ITMS: 26240120029 and VEGA 2/0184/10.

6. REFERENCES

[1] B. Klimt, Y. Yang: Introducing the Enron Corpus. *CEAS, 2004*, http://www.ceas.cc/papers-2004/168.pdf, http://www.cs.cmu.edu/~enron/

[2] C. Bird, A. Gourley, P. Devanbu, M. Gertz, A. Swaminathan: Mining Email Social Networks. In: *MSR '06: Proceedings of the 2006 Workshop on Mining Software Repositories*. ACM, New York (2006) 137–143.

[3] G. Malewicz, M. H. Austern, A. J.C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski. Pregel: a system for large-scale graph processing - "ABSTRACT". In *PODC '09*. ACM, New York, NY, USA, 6-6, 2009, DOI=10.1145/1582716.1582723

[4] M. Laclavík, M. Kvassay, Š. Dlugolinský, L. Hluchý: Use of Email Social Networks for Enterprise Benefit. In: *IWCSN 2010*, IEEE/WIC/ACM *WI-IAT*, 2010, pp 67-70, DOI 10.1109/WI-IAT.2010.126

[5] M. Laclavík, Š. Dlugolinský, M. Šeleng, M. Kvassay, E. Gatial, Z. Balogh, L. Hluchý: Email Analysis and Information Extraction for Enterprise Benefit. In *Computing and Informatics,* 2011, vol. 30, no. 1, p. 57-87.

[6] M. Laclavík, Š. Dlugolinský, M. Kvassay, L. Hluchý: Email Social Network Extraction and Search. In NextMail 2011 workshop, WI-IAT 2011, In The 2011 IEEE/WIC/ACM *International Conference on Web Intelligence and Intelligent Agent Technology*. IEEE Computer Society, 2011, p. 373-376. ISBN 978-0-7695-4513-4

[7] M. Laclavík, M. Šeleng, L. Hluchý: Towards Large Scale Semantic Annotation Built on MapReduce Architecture; In Proceedings of *ICCS 2008*; M. Bubak et al. (Eds.): ICCS 2008, Part III, LNCS 5103, pp. 331-338, 2008.

[8] A. Clauset, C.R. Shalizi, and M.E.J.Newman: Power-law distributions in empirical data. *SIAM Review* 51(4), 661-703 (2009). (arXiv:0706.1062, doi:10.1137/070710111)

[9] J. Judge, M. Sogrin, A. Troussov: Galaxy: IBM Ontological Network Miner. In: Proceedings of the 1st *Conference on Social Semantic Web*, Volume P-113 of Lecture Notes in Informatics (LNI) series (ISSN 16175468, ISBN 9783-88579207-9). (2007)

[10] A. Troussov, D. Parra, and P. Brusilovsky. Spreading Activation Approach to Tag-aware Recommenders: Modeling Similarity on Multidimensional Networks. In: D. Jannach, et al. (eds.) Proceedings of Workshop on *Recommender Systems and the Social Web* at the 2009 ACM conference on Recommender systems, RecSys '09, New York, NY, October 25, 2009.

[11] M. Ciglan, K. Nørvåg: SGDB - Simple graph database optimized for activation spreading computation. Proceedings of *GDM'2010* (in conjunction with DASFAA'2010)

[12] J. Suchal: On Finding Power Method in Spreading Activation Search. In: *SOFSEM 2008: Volume II – Student Research Forum*, 2007, p. 124-130.

[13] J. Suchal, P. Navrat: Full Text Search Engine as Scalable k-Nearest Neighbor Recommendation System. In: *Artificial Intelligence in Theory and Practice III IFIP Advances in Information and Communication Technology*, 2010, Volume 331/2010, 165-173.

[14] M. Ciglan, K. Nørvåg: WikiPop - Personalized Event Detection System Based on Wikipedia Page View Statistics (demo paper), Proceedings of *CIKM'2010*, Toronto, Canada, October 2010.

[15] A. Lumsdaine, D. Gregor, B. Hendrickson, and J. Berry. Challenges in Parallel Graph Processing. *Parallel Processing Letters*, 17(1):5-20, March 2007.

[16] M. Ciglan, A. Averbuch and L. Hluchy: Benchmarking traversal operations over graph databases, Proceedings of *GDM'12, IEEE ICDE Workshop*, 2012

[17] M. E. J. Newman (2003). Mixing patterns in networks. *Physical Review* E 67 (2): 026126.

[18] A. Chapanond, M. S. Krishnamoorthy & B. Yener: Graph Theoretic and Spectral Analysis of Enron Email Data. *Computational & Mathematical Organization Theory*, 11(3), 265-281, 2005

[19] M. Fauscette: The Future of Email Is Social. White Paper; *IBM IDC report*; 2012, ftp://ftp.lotus.com/pub/lotusweb/232546_IDC_Future_of_Mail_is_Social.pdf

[16] http://ikt.ui.sav.sk/esns/enron/

[17] http://gsemsearch.sourceforge.net/

[18] http://www.venis-project.eu/

6.2 Email Social Network Extraction and Search

[LAC11B] LACLAVÍK, Michal - DLUGOLINSKÝ, Štefan - KVASSAY, Marcel - HLUCHÝ, Ladislav. Email social network extraction and search. In The 2011 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology. - Los Alamitos : IEEE Computer Society, 2011, p. 373-376. ISBN 978-0-7695-4513-4.

2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology

Email Social Network Extraction and Search

Michal Laclavík, Štefan Dlugolinský, Marcel Kvassay, Ladislav Hluchý
Institute of Informatics, Slovak Academy of Sciences
michal.laclavik@savba.sk

Abstract

The article discusses our email search prototype, which exploits social networks hidden in email archives and the spread of activation algorithm. The prototype offers new way of searching email archives as knowledge repository. The prototype was partially evaluated on the Enron email corpus.

1. Introduction

Email communication analysis allows the extraction of social networks with links to people, organizations, locations, topics or time. Social Networks included in email archives are becoming increasingly valuable assets in organizations, enterprises and communities, though to date they have been little explored.

Social networks in the area of email communication have been studied to some extent. Communication on the Apache Web Server mailing lists and its relation to CVS activity was studied in [1]. Extracting social networks and contact information from emails and the Web and combining this information is discussed in [2]. Another research effort [3] exploits social networks to identify relations, and tests the proposed approaches on the Enron corpus. We are using a similar approach to that of IBM Galaxy [4] in the Nepomuk[1] project, where the concept of multidimensional social network was introduced. Similar spread of activation inference was also tested on Wikipedia [7].

In the article we present a new approach for email search exploiting information extraction, graphs or networks and spread of activation. We build on our previous work [5] [6], where we presented an approach for the extraction, spreading activation and we also evaluated the relevance of the extraction and inference algorithm with satisfactory results. In this paper we discuss Email Social Network Search prototype, which uses the same algorithm as described in [5] and [6] but a new user interface for controlling the search has been built and we have also performed scalability evaluation on a part of the Enron email corpus[2].

2. Extraction of Social Network Graph

In order to provide the social network graph hidden in the email communication, the important task is to identify objects and object properties in the email and thus to formalize the message content and context. For object identification we use the information extraction (IE) techniques [6] based on regular expression patterns and gazetteers[3] as can be seen in Figure 1.

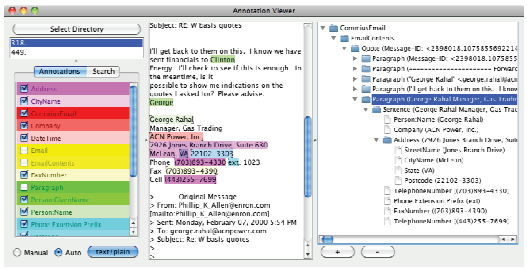


Figure 1. User interface of Ontea [6] information extraction tool with highlighted extracted objects (middle) and tree structure (right) which is used to build social network graph (Figure 2).

Patterns and gazetteers extract key-value pairs (object type – object value) from email textual content as seen in the middle of Figure 1. If there is textual data present in binary form (e.g. PDF attachment) it is, if possible, converted to text before the information extraction begins. Extracted key-value pairs are then used to build the tree (see Figure 1 on the right side) and the graph of social network as can be seen in Figure 2. So the social network contains not only the communicating parties but also related entities, which can be explored.

[1] http://nepomuk.semanticdesktop.org/

[2] http://www.cs.cmu.edu/~enron/

[3] Gazetteer is simply the list of keywords (e.g. list of given names) representing an object type, which are matched against the text of emails.

978-0-7695-4513-4/11 $26.00 © 2011 IEEE
DOI 10.1109/WI-IAT.2011.30

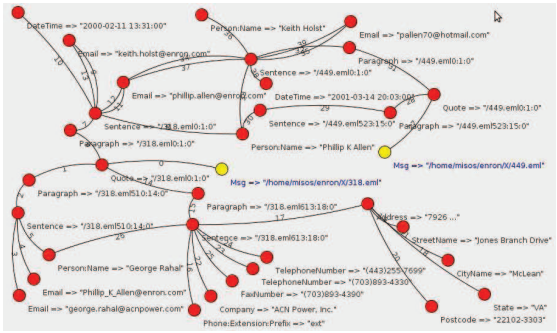


Figure 2. Social Network Graph built from 2 Enron emails. Please note that email or person name connects these 2 emails. Graph[4] also contains sentence and paragraph nodes (Figure 1, right). Here such nodes are hidden for readability.

3. Email Social Network Search Prototype

Email Social Network prototype exploits email analysis and extraction in the form of key-value pairs [6] (Figure 1, middle), semantic trees (Figure 1, right side) as well as their interconnection into a graph of social network [5] (Figure 2). It also exploits the spreading activation inference algorithm discussed in [5].

The idea is that we can infer relations of the activated object using spread of activation in social network graph. For example we can discover related objects for a person as seen in Figure 3.

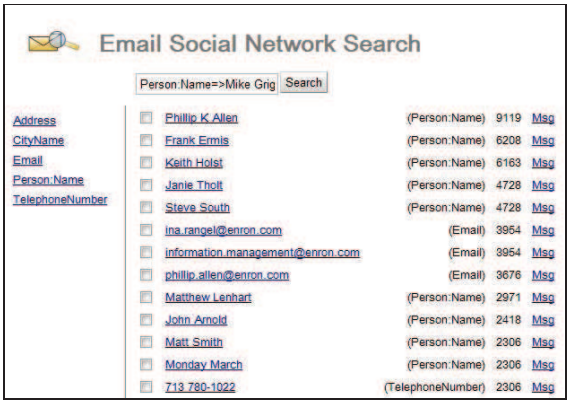


Figure 3. Email Social Network Search user interface. Relevant objects of various types returned for a person.

The Email Social Network Search Module enables the user to search real world objects mentioned in an email or email archive, and their relations. Exploiting the prototype, the user can discover personal email addresses, telephone numbers, company names, organizational names and so on. When the user searches for the object and accesses it, all the objects related to the searched object (i.e. personal names, phone numbers, organizations, addresses) are shown as results and the user can navigate deeper into the object graph by clicking on any object. For instance, the user can select a person's name to get the company where this person works, then select the company to get its phone number or address, and so on.

Figure 3 and 4 shows the user interface of the Email Social Network Search prototype. Its input is a key-value pair representing the business object (in this case, person) extracted from the email. When pressing the search button, the graph node in the search box is activated and the spreading activation algorithm is executed on the social network graph. The algorithm returns all the relevant objects (key-value pairs) for the searched object (Figure 3). For example, it returns organization names, email addresses, postal addresses, telephone numbers or personal names. Figure 4 shows the restriction of the search to return only the objects of a certain type (type is defined by the key of the key-value pair). In this case we wanted to get the phone number of the person, which was quite low in the main (mixed-type) list, so the search was further restricted to return only the relevant phone numbers for this person by clicking on the type *TelephoneNumber* in the left panel of GUI.

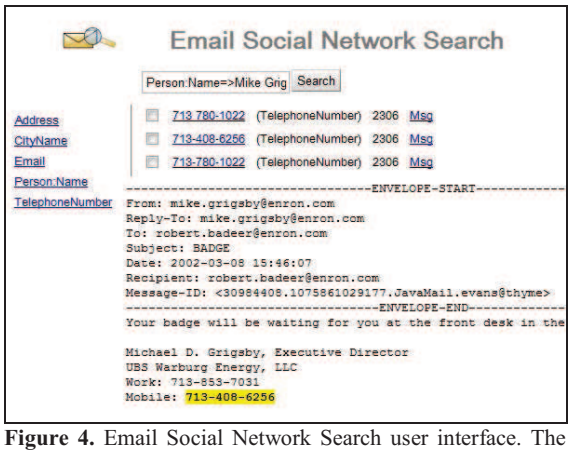


Figure 4. Email Social Network Search user interface. The same results as in Figure 3 restricted to phone numbers only. Phone numbers deemed relevant for the chosen person are displayed. Inferred objects (in this case phone numbers) can also be displayed and highlighted in their "most relevant" email message by clicking on "Msg" link.

In addition we can click on the "Msg" link next to the returned object in order to validate the search result. For example the prototype may return a telephone number of some other person (possibly somehow related) if the searched person does not have

[4] Values on edges are just edge IDs and their values are irrelevant for the algorithms.

any phone in the email archive. The requested object is then highlighted in the returned email message, which is the most relevant email inferred using the spread of activation, because the archive may contain hundreds of messages with the requested object.

The search algorithm can also be improved by allowing the users to delete the wrongly extracted objects or to connect various aliases of the same object (e.g. the same company or personal names spelled differently) as seen in Figure 5. Such a user feedback enables the search to learn and return better results in the future. For example, if we merge 3 selected person name aliases seen in Figure 5, there will be better results (e.g. phone number, address or organization) returned for any of the aliases.

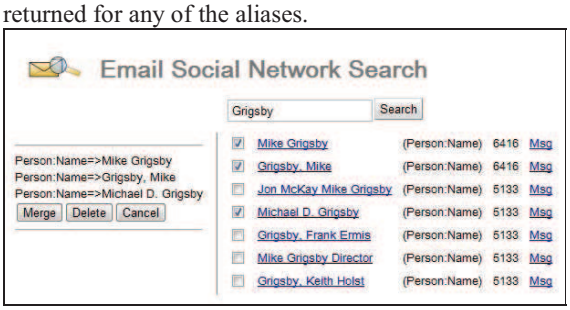


Figure 5. Prototype GUI with results of full-text search and several objects (aliases) selected for possible merge or delete. When these object aliases are merged, subsequent search returns better results for any of them.

In the future, we also plan to extend the prototype to let users annotate their own key-value pairs in the emails (using the Email Social Network Search GUI). In this way they can later easily search for the additional valuable information hidden in the emails, such as passwords, links or documents. In addition the annotated values will also be searched in other emails where they appear. An interesting feature would be to extend the search to include the attachments. We did not do it, since the version of the Enron corpus we have used did not include attachments, but in principle our prototype is now able to identify objects in attachments. The prototype and video explaining the search features is available at http://ikt.ui.sav.sk/esns/ and it works on several Enron mailboxes.

4. Evaluation

In [5] we have evaluated precision and recall of the information extraction and the spreading activation algorithms. The prototype and the algorithm were further refined with focus on higher precision of results. In this section we discuss mainly the scalability of the algorithm. Our hypothesis was that the performance (search time) should be stable even with large graphs, because we always activate only a small portion of the graph. This was found to be valid only to some extent.

Table 1. Email Social Network Search performance evaluation on 5 datasets.

Number of Mailboxes	1	5	7	10	15
Number of Emails	3 033	9 939	20 521	36 532	50 845
Number of Verticles	41812	159 776	369 932	608 146	835 025
Number of Edges	98566	380 254	971 929	1 796 403	2 514 031
Processing time (ms)	81 672	430 025	1 199 463	1 948 847	2 680 171
Processing time (minutes)	1	7	20	32	45
One Email processing time	27	43	58	53	53
Person:Name=>Mike Grigsby					
Search Response Time	**144**	**446**	**758**	**1 396**	**1 696**
Results	344	463	494	781	761
Fired	6 363	20 732	19 045	23 466	23 839
Visited	112 280	281 060	476 324	939 642	1 174 400
Visited Unique	18 382	53 772	82 219	145 192	178 829
Search Slowed down x Times	**1**	**3,1**	**5,3**	**9,7**	**11,8**
Fired x Times	**1**	**3,3**	**3,0**	**3,7**	**3,7**
Number of messages x Times	1	3,3	6,8	12,0	16,8
Number of verticles x Times	1	3,8	8,8	14,5	20,0
Number of edges x Times	1	3,9	9,9	18,2	25,5
TelephoneNumber=>713 780-1022					
Search Response Time	**5**	**8**	**8**	**12**	**13**
Results	4	4	4	4	4
Fired	116	150	157	181	183
Visited	6 318	8 776	9 550	13 424	14 710
Visited Unique	698	954	1 059	1 424	1 513
Search Slowed down x Times	**1**	**1,5**	**1,6**	**2,3**	**2,5**
Fired x Times	**1**	**1,3**	**1,4**	**1,6**	**1,6**
Number of messages x Times	1	3,3	6,8	12,0	16,8
Number of verticles x Times	1	3,8	8,8	14,5	20,0
Number of edges x Times	1	3,9	9,9	18,2	25,5
Address=>6201 Meadow Lake, Houston, TX 77057					
Search Response Time	**7**	**14**	**28**	**40**	**59**
Results	23	38	71	91	170
Fired	236	515	701	896	1 546
Visited	8 134	15 571	32 336	40 563	58 571
Visited Unique	1 097	1 952	6 526	8 029	11 295
Search Slowed down x Times	**1**	**2,1**	**4,3**	**6,0**	**8,9**
Fired x Times	**1**	**2,2**	**3,0**	**3,8**	**6,6**
Number of messages x Times	1	3,3	6,8	12,0	16,8
Number of verticles x Times	1	3,8	8,8	14,5	20,0
Number of edges x Times	1	3,9	9,9	18,2	25,5
Email=>ina.rangel@enron.com					
Search Response Time	**106**	**552**	**1 162**	**2 156**	**3 017**
Results	732	1 764	2 668	2 809	2 952
Fired	5 165	16 062	17 629	19 716	20 997
Visited	91 199	369 584	865 300	1 694 065	2 326 867
Visited Unique	13 355	54 987	81 757	134 876	168 955
Search Slowed down x Times	**1**	**5,2**	**11,0**	**20,3**	**28,5**
Fired x Times	**1**	**3,1**	**3,4**	**3,8**	**4,1**
Number of messages x Times	1	3,3	6,8	12,0	16,8
Number of verticles x Times	1	3,8	8,8	14,5	20,0
Number of edges x Times	1	3,9	9,9	18,2	25,5

One problem is that the social network graph has properties of small world networks. For example, in a similar work performed on Wikipedia graph [7], only two iterations of spreading activation could be performed because otherwise it would visit too many nodes. In [5] we have used 30 iterations, but on large graphs the impact on performance was too high. Now we set up the number of iterations to 4 experimentally. It seems to have no or minimal impact on the relevance of the returned results. The second problem is the implementation of our algorithm, which even with 4

iterations seems to visit too many nodes without firing the value (row *Visited*) and it visits the same nodes several times (compare with row *Visited Unique*).

As can be seen in Table 1, we have tested the search response time on 5 datasets from the Enron corpus. Datasets contained from 3,000 up to 50,000 emails. The biggest dataset resulted in a graph of 800,000 nodes and 2.5 million edges.

Before processing, graphs were loaded in memory. We have used Jung[5] library for graph representation. We plan to test email search also with SGDB [8], which would allow us to work with the whole Enron corpus or even larger email archives and test different versions of the spreading activation algorithm.

We have performed the same searches on all the five datasets for 4 different objects: person, telephone number, address and email address as seen in Table 1. The response time was computed as the average from 3 searches. We expected that the *Search Response Time* would not increase much with the number of nodes in the graph, but this was valid only partially – for the phone number and the address, which were deeper in the graph and had fewer connecting edges. Nodes such as emails or person names appear in emails more often and thus already 4 iterations of spreading activation activate (and especially visit) too many nodes for them. We believe that the problem is also in the implementation of our algorithm. The results show that the counts of the *Fired* and *Visited Uniqe* nodes do not increase so much with the number of nodes and edges in the graph, but the number of *Visited* (processed) nodes increases quite heavily. We believe this can be overcome in a better implementation, which would lead to a significantly better performance. In addition, we probably need to change the number of iterations of spread of activation depending on the topology of the graph near the activated nodes, e.g. to allow less iterations for nodes with many edges.

5. Conclusion

In this paper we discussed new approach to email search exploiting Email Social Networks based on spread of activation and information extraction. We have tested and partially validated our prototype on the Enron email corpus. We believe that the developed search interface, which allows user interaction with social network graph, offers innovative ideas for searching email archives as knowledge repository.

The search response time increases with the number of emails and nodes in the graph in some cases. We have tested it with 5 datasets from 3,000 up to 50,000 emails, where on some objects (person, email) the response time was slow and not so satisfactory, but on other object types such as phone or address the response time kept very good. We have suggested possible improvements, which will be part of our future work. We believe we can improve the response time through fine-tuning the spread of activation algorithm. The prototype and video is available at the web for possible user testing and feedback.

We believe we have shown new innovative approach for email search, which can be connected with data coming from other sources such as transactional databases, web, LinkedData or Wikipedia and thus offer searchable knowledgebase for enterprise, community or personal needs.

[5] http://jung.sourceforge.net/

Acknowledgments

This work is supported by the projects ITMS: 26240220029 and VEGA 2/0184/10.

6. References

[1] Bird, C., Gourley, A., Devanbu, P., Gertz, M., Swaminathan, A., "Mining Email Social Networks", In: *MSR '06: Proceedings of the 2006 Workshop on Mining Software Repositories*. ACM, New York (2006) 137–143.

[2] Culotta, A., Bekkerman, R., McCallum, A.: "Extracting Social Networks and Contact Information from Email and the Web". In: *CEAS'04*. http://www.ceas.cc/papers-2004/176.pdf

[3] Diehl, C. P., Namata, G., Getoor, L., "Relationship Identification for Social Network Discovery" In: *The AAAI 2008 Workshop on Enhanced Messaging, AAAI Conference On Artificial Intelligence*, pp 546-552 (2008)

[4] Judge, J., Sogrin, M., Troussov, A.: "Galaxy: IBM Ontological Network Miner" In: *Proceedings of the 1st Conference on Social Semantic Web*, Volume P-113 of Lecture Notes in In-formatics (LNI) series (ISSN 16175468, ISBN 9783-88579207-9). (2007)

[5] Laclavík M., Kvassay M., Dlugolinský Š., Hluchý L.: Use of Email Social Networks for Enterprise Benefit; In: IWCSN 2010, IEEE/WIC/ACM WI-IAT, 2010, pp 67-70, DOI 10.1109/WI-IAT.2010.126

[6] Laclavík M., Dlugolinský Š., Šeleng M., Kvassay M., Gatial E., Balogh Z., Hluchý L.: Email Analysis and Information Extraction for Enterprise Benefit; In Computing and Informatics, 2011, vol. 30, no. 1, p. 57-87.

[7] Ciglan M., Nørvåg K.: WikiPop - Personalized Event Detection System Based on Wikipedia Page View Statistics (demo paper), Proceedings of CIKM'2010, Toronto, Canada, October 2010.

[8] Ciglan M., Nørvåg K.: SGDB - Simple graph database optimized for activation spreading computation, Proceedings of GDM'2010 (in conjunction with DASFAA'2010)

6.3 Email Analysis and Information Extraction for Enterprise Benefit

[LAC11] LACLAVÍK, Michal - DLUGOLINSKÝ, Štefan - ŠELENG, Martin - KVASSAY, Marcel - GATIAL, Emil - BALOGH, Zoltán - HLUCHÝ, Ladislav. Email analysis and information extraction for enterprise benefit. In Computing and informatics, 2011, vol. 30, no. 1, p. 57-87. (0.356 - IF2010). ISSN 0232-0274.

Computing and Informatics, Vol. 30, 2011, 1001–1031, V 2011-Jan-14

EMAIL ANALYSIS AND INFORMATION EXTRACTION FOR ENTERPRISE BENEFIT

Michal Laclavík, Štefan Dlugolinský, Martin Šeleng
Marcel Kvassay, Emil Gatial, Zoltán Balogh, Ladislav Hluchý

Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9
845 07 Bratislava, Slovakia
e-mail: michal.laclavik@savba.sk

Manuscript received 22 October 2010; revised

Abstract. In spite of rapid advances in multimedia and interactive technologies, enterprise users prefer to battle with email spam and overload rather than lose the benefits of communicating, collaborating and solving business tasks over email. Many aspects of email have significantly improved over time, but its overall integration with the enterprise environment remained practically the same. In this paper we describe and evaluate a light-weight approach to enterprise email communication analysis and information extraction. We provide several use cases exploiting the extracted information, such as the enrichment of emails with relevant contextual information, social network extraction and its subsequent search, creation of semantic objects as well as the relationship between email analysis and information extraction on one hand, and email protocols and email servers on the other. The proposed approach was partially tested on several small and medium enterprises (SMEs) and seems to be promising for enterprise interoperability and collaboration in SMEs that depend on emails to accomplish their daily business tasks.

Keywords: Email, information extraction, trees, graphs, social networks, context, recommendation

1 INTRODUCTION

Recent reports [1] confirm that email is still number one online activity [4]. According to Radicati surveys, corporate users send and receive an average of 110 messages per day [5], out of which about one third are messages sent. These statistics are quite stable and did not change much in the last 5 years [3], but the needed volume for storage and email accounts penetration is increasing, reaching almost 3 billion email accounts over the world (with 1.6 account per user). This includes 730 million corporate accounts, which should grow to almost 1 billion in 2014 [5]. In 2001, corporate users received just about 25 email messages per day and sent about 13 messages [4]. Findings from 2003 [6] also show that 80 % of users prefer email for business communication. Pew/internet report [2] from 2008 says that in the USA, 62 % employees could be considered *Networked Workers*, using the internet or email at their workplace on a regular basis. There is also a growing trend in wireless email accounts including both the enterprise and consumer accounts with the total of about 307 million in 2010, expected to grow rapidly over the next four years to 1.4 billion wireless accounts worldwide [5]. These findings indicate that the importance of email in business may not be adequately addressed in the research.

Information created by business entities can represent an asset or a liability, depending on how well it is managed. Email is not different in this respect: it can be a highly efficient and useful tool for communication, but only if the information it contains can be managed effectively. Email is rarely a standalone information source; it often points to further information such as files (e.g., saved attachments), links to items on the web, and references to other resources. Email is currently used as a conduit for many functions [7, 8], including alerting, archiving, task management, collaboration and interoperability.

1.1 Content Analysis, Extraction and Semantics in Emails

Several existing commercial solutions analyze and partly "understand" the email content. For instance, Gmail focuses mainly on context sensitive advertisements, but can also detect events, addresses or package tracking numbers. Similarly, Zimbra or ClearContext try to recognize some objects in the text. Xobni focuses mainly on the extraction of contact data from email signatures [9, 10]. One of the first attempts to apply Semantic Web technologies to email was performed by McDowell [11], who tried to solve the problems arising in one-to-many communication tasks, such as event planning, by including Semantic Web formal data in the message. One of the most significant attempts to analyse and understand the email communication has been performed with Semanta [12]. It applies speech act theory to the email communication processes, eventually giving a formal structure and semantics to ad-hoc workflows, which are characteristic of email communication. Speech act theory was applied also by [14] for "email acts" classification. For content analysis and semantics, it is also important to have available corpuses with pre-annotated data. Not many of them are available. To our best knowledge, the

only publicly available email corpus with enterprise emails is the Enron corpus [13]. More information on email corpuses, semantics and content analysis is provided in our previous work [9, 10].

1.2 Social Networks in Emails

Email communication analysis allows the extraction of social networks with links to people, organizations, locations, topics or time. Social networks included in the email archives represent a level of semantics beyond speech-acts, and are becoming increasingly valuable assets in organizations, enterprises and communities, though to date they have been little explored. However, email social networks have been studied to some extent. For example, communication on the Apache Web Server mailing lists and its relation to CVS activity was studied in [15]. This work also introduces the problem of identifying email users' aliases. Extracting social networks and contact information from emails and the Web and combining this information is discussed in [16]. Similarly, new email clients, e.g. Postbox or plug-ins Xobni[1], try to connect email social networks with web social networks like LinkedIn or Facebook. Xobni in addition exploits email social networks to help the user manage contacts and attachments. We have also performed some experiments with the extraction of social networks from large email archives and network transformations using a semantic model [17]. A related research effort [18] exploits social networks to identify relations and tests the proposed approaches on the Enron corpus. There is a lot of research focusing on social networks in the context of web social networking applications, but the email social networks are different. In the email archives it is possible to discover the level of interactions (number of messages exchanged, time, relation to content and possibly discovered semantics) that goes beyond what is captured by the current social networking sites, and the potential of these differences for better information and knowledge management still needs to be explored. We are using a similar approach to that of IBM Galaxy [19] in the Nepomuk[2] project, where the concept of multidimensional social network was introduced. In this paper we show the initial results of exploiting the email social network in order to support a better understanding of the email content as well as enabling novel applications such as contact, product, service, partner or supplier search within organizations or communities.

1.3 Contextual Recommendation

Efforts to link emails with knowledge or context-sensitive information have been attempted in several tools [9] such as kMail, Zimbra, Gmail or Xobni. Additional R & D prototypes have been developed to address specific aspects of the general email communication problem (e.g. task management, information archiving, collabora-

[1] `http://www.xobni.com/`
[2] `http://nepomuk.semanticdesktop.org/`

tion, etc.), such as Telenotes, ContactMap, TaskMaster, Snarf, Remail, Priorities or recent Semanta. Our recommendation components build on the extracted information in the form of semantic trees and social networks. Deeper insights into the related work are provided in our previous analysis [9, 10].

1.4 Paper Contribution and Structure

In Section 2 we discuss our approach to information extraction and analysis of email communication as well as its main concepts based on key-value pairs, semantic trees and social networks. In Section 3 we discuss the use cases and prototypes for contextual recommendation and email social network search based on the extracted information. We also discuss Acoma system which can integrate with email infrastructure and describe prototypes and techniques. In evaluation and experiments Section 4 we provide the customization and the relevance evaluation of the approach, as well as the inference evaluation on social networks. Finally we conclude our findings in Section 5.

2 TECHNIQUES FOR INFORMATION EXTRACTION AND ANALYSIS

Information Extraction (IE) techniques [20] usually focus on the five main tasks of information extraction defined by the series of Message Understanding Conferences (MUC):

Named entity recognition (NE) – finding entities. Finds and classifies the names, places, etc.

Coreference resolution (CO) – aliases and pronouns referencing the entities. Discovers the identity relations between entities.

Template element construction (TE) – properties or attributes of entities. Adds descriptive information to NE results (using CO).

Template relation construction (TR) – relations between entities. Finds relations between NE entities.

Scenario template production (ST) – events involving entities. Fits TE and TR results into specified event scenarios.

Several advanced state-of-the-art systems such as GATE [21], KIM [22], C-PANKOW [23] or knowItAll [24] exist and are able to fulfill some of these tasks. C-PANKOW or knowItAll focus on general information domain such as web, and are not applicable to the enterprise-specific content. Usually if we want to get the best results we need to apply Natural Language Processing (NLP) techniques to decompose the sentences, and do Part of Speech Tagging (POS) to distinguish the nouns, pronouns, adjectives and verbs. Then we can apply the techniques of NE on nouns, CO on pronouns and TE on adjectives. The problem occurs if we

want to apply these techniques on languages where the basic NLP support (e.g. stemmers or POS tagging) does not exist. NLP techniques such as those used in knowItAll, C-PANKOW or GATE are applicable mainly to English. Further problems arise if we apply the existing techniques to business documents (e.g. related to interoperability) or email content, which is specific for each enterprise or business sector, and very different from web documents or news articles, where NLP techniques are usually tested. This forced us to explore other approaches to information extraction in the context of business interoperability and email communication.

We found out that a lighter approach – *pattern-* and *gazetteer-based* detection – was not only much simpler, but also easily adaptable to the unique information extraction needs in different enterprises. Patterns are created manually by an expert or skilled programmer familiar with regular expressions. In customization evaluation (see 4.1) we have conducted experiments, which show that the required manual effort is not too high. Pattern- and gazetteer-based extractors generate key-value pairs (object type – object value), that are used to form semantic trees (see 2.2). It is possible to connect the extracted pairs and trees into a so-called multidimensional social network representing the email communication. Such semantic graphs or networks can be exploited to return the relevant results and discover the relations among business objects by relatively simple algorithms, such as spread of activation. We discuss this in Section 2.3.

Although we prefer the pattern-based detection as a light-weight and flexible approach, our general software framework can integrate any information extraction tool (including NLP), so long as it provides key-value pairs as output. In this respect, we have for instance succeeded in integrating our prototype with the GATE system and also used the Ontotext standalone gazetteer[3] originally developed for GATE.

Email communication typically lacks strict structure, but in many cases it carries structured or semi-structured information. This applies to business communication and especially to interoperability (or transaction) emails. Such emails typically contain business data or documents with objects and properties such as company names, amounts, product codes, bank account numbers, product quantities or prices. As already mentioned, the state-of-the-art IE techniques are usually meant for web documents or news. Email is different in the sense that email archives often contain references to enterprise-specific business tasks, processes, products, services or transactions (often time-related) and threads of communication. Social networks of interacting people can be extracted from the emails as well. They may be enriched with the related business objects, thus giving rise to multi-dimensional social networks. Our approach is based on the idea that a substantial portion of this valuable information can be extracted using the light-weight pattern-based approach in combination with gazetteers to fulfil the following IE tasks:

[3] `http://www.ontotext.com/downloads/index.html`

NE: entity detection

CO: aliases detection (e.g. product by detecting the product code, or customer by detecting his/her email or phone)

TE: attributes and properties by the segmentation of messages and grouping of properties or using relations in the email messages

TR: relations between entities gathered from external systems (e.g. email-customer) but also relations based on multidimensional social network extracted from email communication

ST: mainly related to email itself, time of sending and the related tasks or transactions.

The extracted data goes beyond the entities and their properties; it includes the relationship graph as well. It can be further analyzed for a more precise interpretation of the intent of the messages, and the results passed on to new tools assisting the enterprises in their business tasks. In this paper we show several possible ways how to use the extracted information.

In addition to the pattern-based extraction, we detect entities (NE task) using gazetteers. Gazetteers are lists of objects of concrete type represented by strings that can be matched in the text. The well-known gazetteers from the existing IE tools are gazetteers for geographical location names, lists of organizations or people. In the business context the most important are the gazetteers representing products, services, customers or suppliers. Such lists can be created from the existing legacy systems in organizations but can also be partially extracted from the email archives using predefined patterns. We have been developing and improving our information extraction approach over the past few years. Our tool, Ontea, is being continuously developed as an open source project[4]. User interface of the recent Ontea version, which focuses primarily on emails, is shown in Figure 1.

The power of the Ontea approach is in its simplicity [25] (compared to more advanced but heavy solutions such as GATE), as well as in its ability to define transformation chains and to connect to information system environment (databases, documents, intranets, and internet). In addition, Ontea supports email decomposition and analysis of email header, body and attachments. Supported attachment types are emails in .eml format, MS Word, PDF and text files, which are converted into a plain text. After an attachment or its content (archive file attachment) is converted to plain text, it is further processed like text in email.

We have tested our approach in the context of 6 organizations. Within the Commius project[5], we have tested it on the emails from Softeco, Aitek and Techfin SMEs (Italy), and from the Fedit technology center (Spain). Our relevance and social network evaluation in Section 4.2 is based on Fedit emails. In a related AIIA[6]

[4] `http://ontea.sourceforge.net/`
[5] `http://www.commius.eu/`
[6] `http://aiia.ui.sav.sk/`

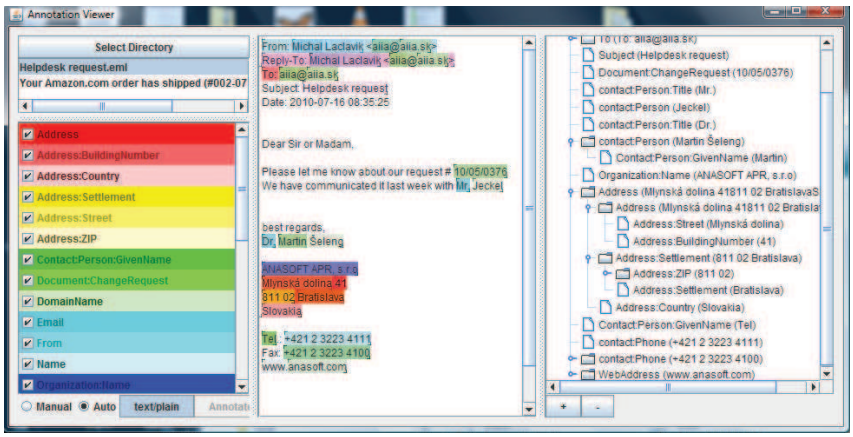


Fig. 1. Email message processed by the Ontea information extraction tool. Detected objects are highlighted. On the right, several objects such as address or person are grouped into a hierarchical tree.

Slovak national project, we have tested our information extraction on the emails of two organizations: the first was an SME (Anasoft), the second an academic internet provider (SANET). These experiments showed that *pattern-* and *gazetteer-based* IE approach can be customized for a specific application area (where the objects need to be discovered in the email communication) in a few hours.

Thus the aim of the Ontea IE is to create and share the patterns and gazetteers for objects and object properties. We believe the best approach for IE in the enterprise context is the pattern- and gazetteer-based extraction, for several reasons:

- Patterns can be adapted for enterprise business needs. For example, the formats of product codes may differ between companies, but within one company they tend to be well-defined and adhered to.
- Patterns can be defined, improved, evaluated and shared for a group of SMEs or for a community around a similar business model or industry type. This role is best suited for IT providers of these SMEs and communities.
- Patterns are easily adaptable to different languages when compared to advanced NLP tools, since NLP requires lemmatizators, stemmers or Part of Speech (POS) tagging tools, which are not available for many languages.
- Emails naturally contain many objects easily extractable by patterns, such as email addresses, phone numbers, people names, company names, dates, websites, addresses or other contact details. In addition, social networks, interactions and message passing data can be extracted from email headers using patterns.
- Business content usually consists of many objects and properties that can be similarly extracted via regular expression patterns, such as amounts, product codes, bank account numbers or customers.

- Patterns can be combined together with gazetteer results. For example, we have used this approach on person and company names extraction, where we defined a gazetteer list for given names, company legal classification abbreviations as well as for some common company-name specific words like Bank of, Hotel, Association, etc.

2.1 Key-Value Pair-Based Information Extraction

Early results of our approach were already presented in [25], where we also presented various information retrieval techniques for aliases and disambiguation as well as key-value transformations and integration with a variety of tools. We have also provided the state of the art of the relevant information extraction and semantic annotation techniques.

The main concept is key-value pairs representing object type (key) and value (matched text). For example, in Figure 1 we can see key-value pair, where the key is *contact:Person* and value is *Martin Šeleng*, which was extracted by the regular expression pattern of two words starting with capital letters (in this case also by identifying the title *Dr.* at the beginning). Please note that there are also other key-value pairs like *contact:Person:GivenName – Martin*, which are sub-parts of previous key-value pair. This key-value pair was extracted by the gazetteer for given names. We extract all the key-value pair candidates using the predefined patterns or gazetteers independently, often maybe identifying the same key-value pair by several techniques. Positions of key-value pairs are known and can be used for further processing, for example deleting the contact:Person pair candidate if GivenName is not matched inside. Here is an example of such configuration in Ontea tool:

```
ontea.core.gazetteer.Gazetteer {  gazetteer/lists.def }
ontea.core.xmlregex.XMLRegexExtractor {  patterns/person.xml }
ontea.transform.resultset.RuleTransformer {
  contact:Person* =>
  contact:Person:GivenName =>
  contact:Person:GivenName * => contact:Person:GivenName
}
```

We will not explain the above example in detail, but as you can see, gazetteer, pattern extractor and key-value set transformation is configured. Each block in configuration starts with Java class name implementing extraction or transformation interfaces and the brackets{} contain input for the implemented class. The Java classes are loaded dynamically by name, which allows to add more extraction and transformation tools into the system as plug-ins implementations of the interfaces. Similarly the Gate application can be plugged-in.

The example below shows fragment of patterns for the detection of postal address, which define macros for various address parts, such as city or zip, which are then incorporated into the postal address pattern. Macros can be reused. Based on

the overlap of the extracted key-value pairs, we can build the trees, which can be seen on the right side of Figure 1 and in Figure 2. Trees are discussed in the next section.

```
<?xml version="1.0" encoding="UTF-8"?>
<patterns>
  <pattern name="Postcode" class="Postcode">
    <regexp><![CDATA[(?:[0-9]{3} *[0-9]{2})]]></regexp>
  </pattern>
  <pattern name="CityName" class="CityName">
    <regexp><![CDATA[(?:\p{Lu}\p{Ll}+(?: \p{L}+)*)]]></regexp>
  </pattern>
  ....
  <pattern name="PostcodeCityLine">
    <regexp>
      <![CDATA[(?:\p{pattern:Postcode}+\p{pattern:CityName}
        (?: +\(\p{pattern:CountryName}\))?)]]>
    </regexp>
  </pattern>
    ....
</patterns>
```

2.2 Semantic Trees

Our approach to building semantic trees is quite novel and is not mentioned in the state of the art information on the extraction tools elsewhere. The extracted key-value pairs are formed into hierarchical trees depending on their positions in the text and the key names logical dependencies according to a pre-defined XML schema and tree transformation rules. Hierarchical trees give us extra information about the relationships among the result instances. For instance, an Organisation result is a sub tree that contains nested results like Name, Email, TelephoneNumber, StreetName, BuildingNumber, PostCode, CityName, etc. From the tree we can see that its sub-components are related to each other since they have a common parent (Figure 2 left).

In the Commius project, we use modified Core Components specification[7] to represent the information extraction results; but constructing a result tree compliant to CoreComponents specification does not involve just the information extraction. It also requires subsequent transformations of the hierarchical tree of results, for which we use several tree transformers. We define tree transformation rules, which call the tree transformers and can create, delete, move, rename or reorder the tree nodes (information extraction results). The tree on left side of the Figure 2 represents the Organization object compliant to the Core Components XML schema. Such XML

[7] http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf

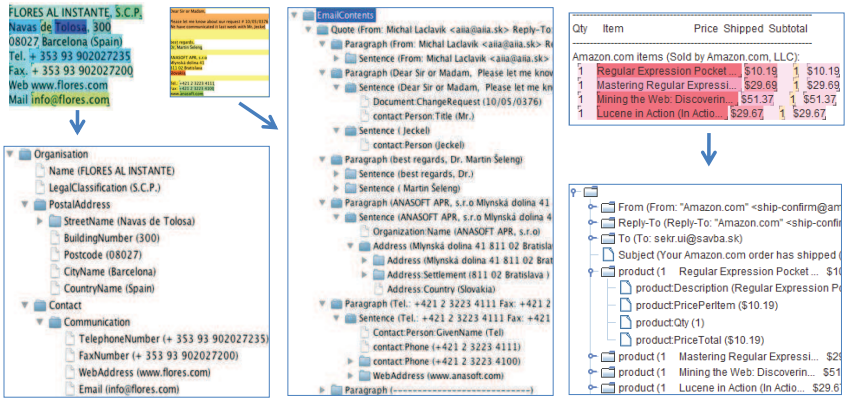


Fig. 2. Left: An example of a hierarchical sub-tree of information extraction results with a corresponding source text. The hierarchy is determined by the result positions and tree transformation rules, as well as the extraction pattern nesting. The result is based on Spanish email from Fedit application. Middle: An example of a hierarchical tree built upon email segmenter results. Email text is divided into quotes, paragraphs and sentences. Example is based on the same email as seen in Figure 1. Right: The Amazon shipment email processed by the Ontea tool. We can see the generated product tree.

objects are then used in Commius to identify document types [28] and determine the current processes [29].

We build hierarchical trees from the ordered lists of results. Results in the lists are ordered by their position relative to the beginning of the source text and by their length (or end position). The length of the result is important when sorting the list items, because we can decide whether one result contains another one and put the parent result before its child results in the list. If we have such ordered list, then it is simple to produce a hierarchical tree from it. The basic hierarchy of results can be acquired from the results produced by email segmenters. We use several simple segmenters such as email quotation segmenter, paragraph segmenter and sentence segmenter. These segmenters divide email text into structured segments as information extraction results from which we can build a hierarchical tree (Figure 2 in the middle). Segmenters make a hierarchy starting up with a quotation block, followed by paragraphs, sentences and results. This hierarchy describes the relation of results on the basis of their closeness to each other, where we assume that results found in a sentence are semantically more closer than results in a paragraph. Information extraction results produced by the segmenters are exploited in recommendation prototype (Section 3.1) as well as in the social network inference algorithm described in Section 2.3.

Moreover, the hierarchy of the results in the tree is also revealed by the key-value pair results. This is done by exploiting the information extraction model we use.

The extraction model lets us create complex extraction patterns consisting of other patterns/macros, which together represent a composite structure. A good example is the postal address pattern, where we have several macros defined for street name, street number, city name, etc. and we use them in the address pattern (see patterns example in Section 2.1).

We are able to build rich objects using our tree- and graph-oriented approach and the transformation techniques described above. Complex structures such as those shown in Figure 2 can be transformed into XML according to standards such as Core Components, but can also be used to create rich ontology instances and thus serve not only the needs of information extraction but for semantic annotation [25] as well. One semantic tree represents one email message, but when we connect the trees from the whole email archive (they are interconnected through shared objects), then we get a multi-dimensional social network or graph, which can be used for knowledge management or for inferring hidden relationships, as discussed in the next section.

2.3 Email Social Networks and Graph Inference

Information Extractor (IE) primarily handles the straightforward cases of structured objects like postal address where the sub-components (street name, ZIP code, city, etc.) immediately follow one another in a predefined sequence captured by regular expressions and macros. Graph inference is meant for the tasks beyond the reach of this method. One such example is to gauge the relationship between the telephone numbers and people which, in general, need not immediately follow one another in the email text. The task to assign telephone numbers to people is then a form of uncertain inference, which we are accomplishing through the application of spreading activation to the multidimensional social network graph provided by the information extractor.

We implemented our *social network extractor and analyzer* in Java on top of the information extraction tool Ontea and open-source graphical library Jung[8]. The novelty of our approach is in the application of the spreading activation algorithm to the twin tasks of reconstructing the social network from emails, and then efficiently searching the social graph. The prototype implementation described below is a work in progress. The evaluation of our initial experiments is provided in Section 4.3.

In [26] we have introduced two approaches to building multidimensional social network graphs depending on the format of the information extraction results. Here we only use the more advanced one where IE extracts complex objects (e.g. an address) consisting of simpler objects (e.g. a street name and a ZIP code), and preserves the information about their physical proximity by building a hierarchical tree that includes the nodes representing the sentences, paragraphs and blocks of the message in which they were found. Graph is built from hierarchical tree that can be seen in Figure 1 in the right column or in Figure 2. If the same object (string) is

[8] `http://jung.sourceforge.net/`

found in several messages, it is connected to all of them, but represented as a single object (node) in the graph. If the same object is found multiple times in the same message, it has multiple links to that particular message (or to its parts), which was our way of recording the strength of the bond.

2.3.1 Cumulative Edge Scorer with Attenuation

The structure depicted in Figure 3 can be visualized as a multipartite graph provided the objects of each data type are considered a separate partition. The objects in each partition have no connections among themselves, only to objects in other partitions, which is advantageous from the point of view of computational complexity. For the reconstruction of social network, our prototype *Cumulative Edge Scorer with Attenuation* implements a simplified breadth-first variant of spreading activation. The reconstruction consists in assigning the potential attributes, such as postal addresses, telephone numbers, etc. to the identified primary entities – people or organizations. For the purpose of testing our prototype we have chosen the task of assigning telephone numbers to people.

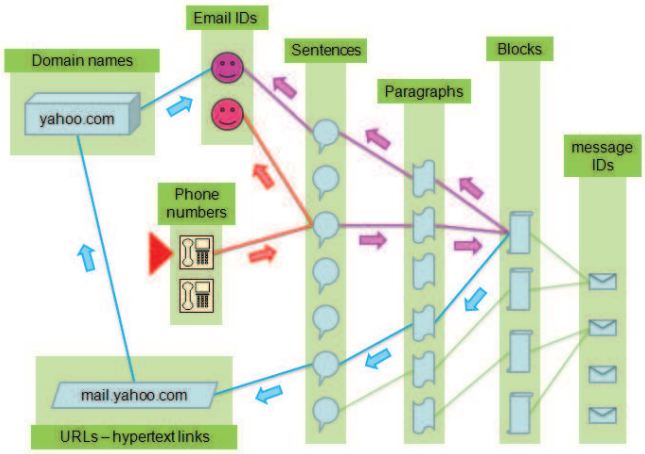


Fig. 3. Spreading activation in a multipartite graph. It starts at one attribute instance (phone number) indicated by the red arrow (center left) and flows towards candidate primary entities (email IDs) by a variety of ways. Since the activation gets attenuated each time it passes through an edge, the shortest path will carry over the greatest increment. But the longer paths can be so numerous that – depending on the value of attenuation and other parameters – their accumulated contribution will ultimately prevail

We were able to simplify our implementation of spreading activation by letting each node fire only once. This was possible because in our case the spreading activation always started from a single node – the attribute instance that we were

trying to assign to a primary entity. Similarly, we did not restrict the maximum value of the activation value that a node could accumulate. Such a restriction is natural and appropriate e.g. in neural networks, but in our case it would be counterproductive since we were using the accumulated activation as a score to determine the most likely "owner" of the attribute instance. The result of these simplifications is an elegant and compact procedure described by the following JAVA-based pseudo-code.

Breadth-first variant of Spreading Activation. *Current_wave* collection (implemented as a *HashMap* with the node as key and its *accumulated_activation* as value) contains the nodes that are going to fire. At the beginning, it only contains one node – the attribute instance that we are trying to assign to a primary entity. In the next step, all its neighbouring nodes are placed into the *new_wave* collection as candidates for firing in the next iteration, and their accumulated activation is incremented by the value of *activation*, which reflects both the edge attenuation and the degree of the firing node. The iteration is completed when all the neighbours of all the nodes in the *current_wave* are processed. After each iteration, the number of iterations is incremented and the nodes in the *current_wave* are transferred into the fired collection so they do not fire again. The *current_wave* is then refilled by the *new_wave.get_activated_vertices()* method that returns the nodes meeting the activation criteria. The nodes representing the primary entities never fire. They accumulate in the *new_wave* and are returned at the end of the algorithm by the *get_output_type_vertices()* method. Their accumulated activation becomes their score, and the primary entity with the highest score will "own" the attribute instance.

```
do
  for (Vertex v : current_wave)
    activation = v.accumulated_activation();
    activation = activation/(attenuation * v.degree());
    for (Edge e : v.outgoingEdges())
      Vertex w = neighbouring_vertex(v,e);
      if (!has_fired(w))
        new_wave.smart_add(w, activation);
  total_iterations++;
  fired.add(current_wave);
  current_wave = new_wave.get_activated_vertices();
  new_wave.remove_activated_vertices();
while (!done());
return  new_wave.get_output_type_vertices();
```

The resulting scores of primary entities for each attribute instance (internally stored in nested HashMaps) can be output in the form of a hierarchical XML. This XML can be converted to HTML by XSL transformations. Results of spreading activation algorithm are also exploited in Email Social Network Prototype discussed in Section 3.2 and shown in Figure 6.

3 PROTOTYPES AND USE CASES

In this section we discuss the Acoma system, which integrates email analysis and information extraction approaches with recommendation and email social network search. We also describe prototypes for contextual recommendation and email social network search on simple use cases.

3.1 Contextual Recommendation Prototype and Use Case

We have developed simple contextual recommendation, built on top of the extracted data. The data needed for the recommender are in the form of extracted key-value pairs formed into semantic trees discussed in Section 2.2. For this kind of recommendation we do not use social networks or graph data, which are used in the next use case. Context for the recommender is key-value and tree based representation of email message. When the information extraction results are available (Figure 1), the email is further processed and additional information is added to the message in the form of HTML links or text attachments (see Figure 4). These additions contain further relevant information and knowledge, hints or links to business resources such as document repositories, databases or information systems needed in the detected business context. When checking for new emails, the users may receive added information/hints either in the modified (enriched) email as an attachment to the original unmodified message (this applies mainly to mobile devices), or in the original message as an attachment to the enriched HTML email with the additional information as seen in Figure 4. In this way the users can configure the email enrichment process according to their needs depending on device, email client or user preferences.

Our solution can successfully process mainly formal, system-generated emails such as Amazon shipment in Figure 2 on the right side. Formal emails are present in many cases of system-to-person communication, for example when purchasing or ordering goods and services on the Internet, in case of transaction status emails or transaction notifications. Typical examples are bank statements, hotel or air-ticket reservation confirmations, invoices to be paid, or shipped goods notifications. Such formal emails with fixed structure are easier to analyze and enrich with context-relevant information. They are often used in business tasks of SMEs and currently need to be processed manually, while we are able to process them semi-automatically. Our experiments, such as the example in Figure 4, have demonstrated that even regular person-to-person emails contain many structural parts and objects which can be detected using patterns or gazetteers.

Our solution, which connects to the email infrastructure and enriches the messages according to the context provided by the information extraction, is called Acoma [31] and is developed as an open source project[9]. Its internal architecture and modularity is described in Section 3.3.

[9] `http://acoma.sourceforge.net/`

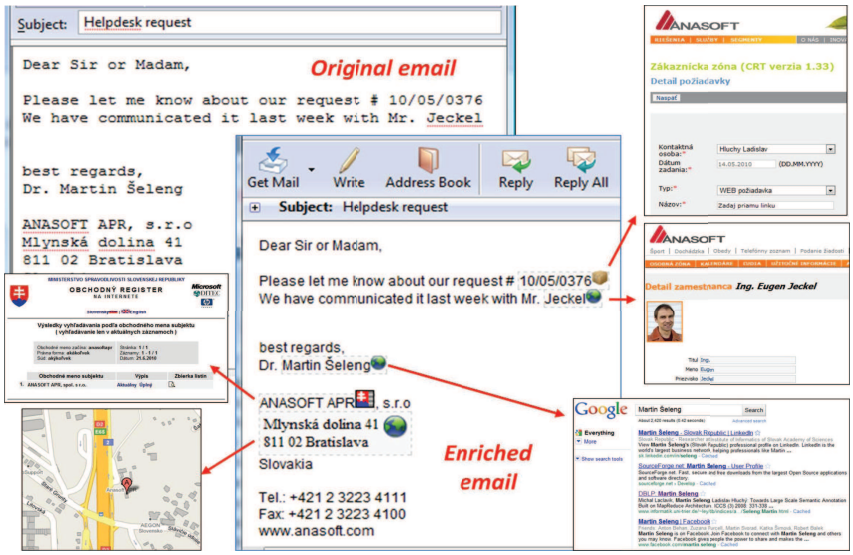


Fig. 4. The same email as in Figure 1 is now enriched and changed into an HTML message with recommendations and links that can help process the email and the tasks it represents. For example, the user may go to the commercial register to find further information about the company, see its address on the map, or search for person in Google or some legacy system such as CRM.

3.1.1 Example of Use

A sample email processed by the Acoma system can be seen in Figure 4. It is based on a modified Anasoft Helpdesk request visible in Figure 1 (we changed the private info and translated the email into English). Acoma replaced the original plain text message by an HTML message including the recommendation hints with links. The links can be placed either inside the message text or on the right side of the message depending on the configuration. (The original email is attached and available to the user if needed.) Thus, the email message contains suggestions for actions to be taken to process the email.

Actions such as accessing a help desk request or a person in a legacy system, showing an address on a map or accessing company information in the commercial registry can be performed by clicking on the recommended hint represented by a box with icon.

The modified email also contains the link that will display the Graphical User Interface (GUI) related to the message in the browser. The GUI contains similar or extended information related to hints and provides the dynamic and interactive functionality, which cannot be achieved by the static HTML included in the email.

It can also integrate the legacy systems for which web GUI is not available. GUI offers extensive functionality but is not discussed in this paper.

Depending on the settings and the email client, Acoma can modify the email messages to include the text, html attachments or just a link to the message GUI.

3.1.2 Recommendation Hints

Information extraction results provide the necessary context for recommendation. Recommendation subsystem is quite simple. It just matches and fills in the text of hints with key-value pairs discovered by IE. It iterates over predefined hint templates and if the required key-value pair (keys are in {} brackets) for a hint is found, the hint template is filled in with the extracted information (value of the key-value pair). The results are then passed on for message post-processing to be included in the email as HTML-formatted output or simple text attachment with links.

Table 1 lists hint templates; in Table 2 the hints are replaced by the key-value pairs extracted from the email using the Ontea IE discussed in Sections 2 and 2.1. Hint templates can be defined through special user interfaces (Figure 5), where the same information as that available in Table 1 can be defined using the GUI.

Fig. 5. Interface for editing generic hint templates, with *See address on the map* hint template displayed. Objects that have to be found in order to fire the hint are selected on the right.

The first hint *show address on the map* is more complex than others, because it requires three objects to be found in the text of the message (street, settlement and zip). This hint is also shown in Figure 5. The problem arises if more than one street or one city name are discovered in the text of the email. In that case we first need to decide, which street and city belong to which address. We are solving this by email segmentation and trees (Section 2.2), and the required objects had to appear only once in common subtree, which can be paragraph, sentence or complex object in tree structure. If we will find all the required objects within one sub-tree (see Figure 1, for the address sub-tree, middle part of the Figure 2 for sentence or paragraph subtree or right part of Figure 2 for product sub-trees) the hint is fired. If we cannot locate the single required key-value pairs for a given hint template within one common sub-tree, the hint is not fired.

Hint Templates
See {address:Street}, {address:ZIP} {address:Settlement} in Google Maps http://.../maps?q={address:Street},+{address:ZIP}+{address:Settlement} address:Street, address:ZIP, address:Settlement link
Organization {org:Name} in commercial register http://orsr.sk/hladaj_subjekt.asp?OBMENO={org:Name}&PF=0&SID=0&R=on org:Name orsr
See request # {doc:ChangeRequest} http://www.ana.sk/crt/main/task_detail.aspx?taskId={doc:ChangeRequest} doc:ChangeRequest product
See person: {person:Name} http://www.google.com/search?q={person:Name} person:Name link

Table 1. List of hint templates used in the example shown in Figure 4. First line: text of hint; second line: link (URL); third line: objects to be found in the text in order to match the hint; last line: hint type/icon. Text in brackets {} are object types (keys) detected by the IE and replaced by concrete values in the generated hints.

Generated Hints
See Mlynská dolina 41, 811 02 Bratislava in Google Maps http://...google.com/maps?q=Mlynsk%C3%A1+dolina+41,+811+02+Bratislava link, 174, 209
Organization ANASOFT APR in commercial register http://orsr.sk/...asp?OBMENO=ANASOFT+APR&PF=0&SID=0&R=on orsr,155,166
See request # 10/04/0647 http://www.ana.sk/crt/main/task_detail.aspx?taskId=43048 product,59,69
See person: Martin Šeleng http://www.google.com/search?q=Martin+%C5%A0eleng link,140,153 See person: Jeckel https://intranet.ana.sk/.../DispForm.aspx?ID=61 link,113,119

Table 2. List of generated hints based on the templates from Table 1. First line: text of hint; second line: link (URL) to external system; third line: hint type/icon, start and end position in text.

Table 2 shows the hints generated from the hint templates in Table 1. The first hint is the already mentioned multi-object match hint, which can be used for *show address on map*, *process transaction* or *manage contact details* user actions. The second hint is the simplest hint requiring only one object (organization name) in the text; the same name is then reused for generating the link to the commercial registry. The third hint is again related to a single detected object, *change request ID* from the *Anasoft Helpdesk* application, but please note that while we are detecting the ID *10/04/0647* in the text, to access the intranet system we need to convert this ID value into *43048* in order to generate the appropriate URL. This is done by an extension, which is not described in this paper but uses the idea of key-value pair transformations mentioned in Section 2.1 and described previously in [25]. Similarly, the last hint appears twice in the email, since two people were detected. If a person is detected as an Anasoft employee, the link to his/her Anasoft intranet card is generated, otherwise just a link to Google Search is offered.

3.2 Email Social Networks Search Prototype and Use Case

Email Social Network prototype can be also integrated as a module into Acoma system, where it exploits email analysis and extraction in form of key-value pairs, semantic trees as well as their interconnection into graph of social network. It exploits the spreading activation inference algorithm discussed in Section 2.3.

Email Social Network Search Module enables the user to search real world objects mentioned in an email or email archive, and their relations. Exploiting the module the user can discover personal email addresses, telephone numbers, company names, company products and so on. When the user accesses the module GUI, all the objects related to the current email message (i.e. people, products, contact details) are shown as results and the user can navigate deeper into the object graph by clicking on any object. The user can, for instance, select a persons name to get the company where this person works, then select the company to get its products, and so on.

Figure 6 shows the user interface of the Email Social Network Search module. Its input is a key-value pair representing the business object (in this case, *person*) extracted from the email. When pressing the search button, the spreading activation algorithm is activated on the social network graph. The algorithm returns all the relevant objects (key-value pairs) for the searched object. For example, it returns organization names, email addresses, postal addresses, telephone numbers and locations. The left part of the screenshot shows the restriction of the search to return only the objects of a certain type (type is defined by the key of the key-value pair). In this case we wanted to get the phone number of the person *Enrico Morten*, which was quite low in the main (mixed-type) list, so the search was further restricted to return only the relevant phone numbers for this person by clicking on the type *TelephoneNumber* in the left panel of GUI.

The search algorithm can also be improved by allowing the users to delete the wrongly extracted objects or to connect the various aliases of the same object (e.g.

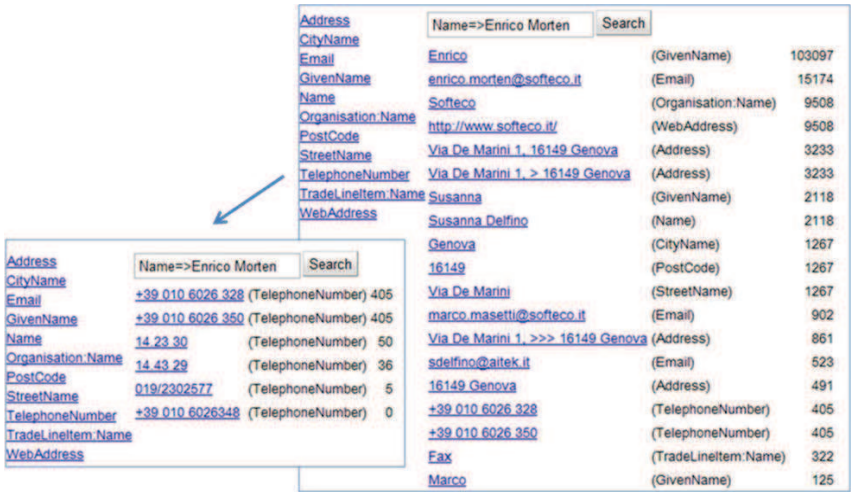


Fig. 6. Email Social Network Search prototype GUI

the same company, personal names spelled differently, or the same address *Via de Marini* on the screen). Such user feedback will enable the module to learn and return better results in future. It is also possible to let the users annotate their own key-value pairs in the emails. In this way they can later easily search for the additional valuable information hidden in the emails, such as passwords, links or documents. An interesting feature would also be to extend the search to include the attachments.

3.3 Integration with Email Environment

The Acoma tool integrates the information extraction, contextual hint recommendation and the enrichment of the messages with email protocols and email servers. Moreover, the information extraction, recommendation and other functionality can be extended by implementing new modules which take email as input and produce recommendation hints as output. In a similar way, the hints shown in Figure 4 and described in the previous section were generated by three separate modules: generic hint template module (1st and 2nd hint), key-value pair transformation module (3rd hint) and a combination of key-value pair and URL transformation module.

Acoma architecture can be seen in Figure 7. The ambition is to support users in business tasks in the context of email communication. We do not want to force people to use new webmail interfaces, new plug-ins to desktop email clients or new email clients, but rather allow users to send and receive emails as they are used to doing it. The Acoma system is hooked to the mail server or desktop in a similar way as email antivirus programs (Figure 7). In this way it can be used with any email

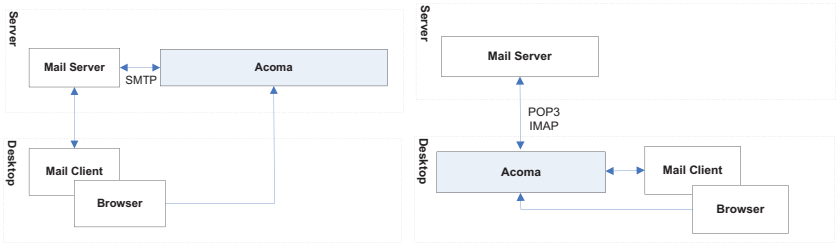


Fig. 7. Acoma on the server (left) and on the desktop (right)

client or even mobile device, without requiring changes to working practices or the adoption of new tools. By a combination of server and desktop use, the users can ensure that security and privacy issues are taken into account. Users and providers can be aware of what data is shared and passed via the communication.

3.3.1 Adaptability

Acoma is implemented using the OSGi[10] approach, which is one of the standards for modular systems in Java. We are using the Apache Felix[11] implementation of OSGi. This enables us to adapt the solution to integrate different functionalities such as context sensitive recommendation or Email Social Network Search into one integrated system accessible from user emails. We believe that such an email based system can benefit from the community approach: just as people are creating applications for Facebook or modules for different open source software, Acoma users can create, modify and share new modules, which will be able to execute or support specific email activities (for example, the Geo module for identifying and displaying the addresses on a map as shown in Figure 4). Other module examples include package tracking, event detection and adding events to calendar, Email Social Network Search module or modules for processing bank statements from a specific bank, service provider or mobile operator. This way the user can just download and add new modules to their Acoma system based on their needs, and so benefit from additional functionality. Even the service providers can create and share Acoma modules to their benefit. Users can be notified on module availability and install them upon receiving the notification. Within 6 SMEs which we have partially evaluated, we have identified the following modules: Catch a contact; Identification of the responsible person (related also to Email Social Network Search module); Customer card (CRM); Geo module; Voting module; Generic hint module; Generic key-value transformation and URL transformation modules. In addition, partner search module, attachment manager and mail template module are being developed in the scope of Commius project.

[10] `http://www.osgi.org/`
[11] `http://felix.apache.org/`

Voting, Geo and Email Social Network Search modules are implemented as prototypes. For the rest of the modules we have tested IE on emails of one or more enterprises. In this paper (Figure 4) we used the Geo module and early prototypes of simple modules. Hint templates presented in Section 3.1 will be covered by the final prototype of the generic hint module in future. When all the needed objects are detected, the hint action can be defined by an URL pointing to some existing web-based system. Functionality for dozens of hints can be created and implemented in a few hours. Customization of the information extraction is more difficult (Section 4.1) but can also be accomplished in a reasonable time-frame. We discuss this in the next section.

4 EVALUATION EXPERIMENTS

In this section we describe the relevance and customization evaluation of our Information Extraction approach. We also discuss experiments on social networks formed from the hierarchical trees and the results of the spread of activation inference in such networks. We do not evaluate the system as a whole, but we evaluate just main email analysis and information extraction approaches described in Section 2.

- Since building of patterns for key-value pair extraction is a manual process, we evaluate the customization process of key-value pair extraction. See Section 4.1.
- We also evaluate success rate of such extraction on selected object types such as people names, addresses and telephone numbers. See Section 4.2.
- Semantic trees are not directly evaluated. They have been found helpful in contextual recommendation and they are also needed in social network relation inference. Its evaluation is given in Section 4.3.

4.1 Information Extraction Customization and Experiments

In this section we describe how the proposed approach for IE can be customized for a specific enterprise or application. We have conducted several experiments in the Commius and AIIA projects, where these information extraction tools are being extended and developed.

The AIIA project focuses on two applications: Anasoft helpdesk, where the products, modules and components mentioned in the email are identified, and an appropriate contact person within the organization is chosen to deal with the email. Automatic identification of the customer contact details is performed in order to retrieve the CRM information related to the customer or change request. The second application is in SANET – the academic internet provider. The focus is on document- and task-related voting through email, in which the contacts, people, agreements and documents need to be identified.

The Commius project focuses on the enterprise interoperability for SMEs. It is important to extract organizations, people, products or transactions from the orders and invoices transmitted via email.

In the first experiment in Anasoft, we have annotated Helpdesk emails. Manual annotations were created with the Ontea tool, which supports this kind of task. Tags were created by the authors of this article (2 people), company expert (1 person) as well as real Helpdesk worker (1 person). We have selected 15 emails where 93 tags were created.

Manual annotations were also tested in the Commius project where several annotation events were organized in Italy and Spain. In Italy Softeco, Aitek and Techfin SMEs were involved. In Spain, Fedit technology center was involved. The focus of Commius annotation events was on the business aspects of perceiving and decomposing objects as well as on their mapping on UN/CEFACT Core Component[12] interoperability standard. Results are available in the Commius deliverable [27] and also in an article in this CAI journal issue [28]. The first round of experiments led us to the definition of the following customization process and to changes in the Ontea tool in order to support it. We believe the process is valid for any enterprise application where IE is needed:

1. *Emails/files browsing*, the step performed by the application users and the developer.
2. *Definition of the objects types and properties to be extracted*, the step performed by the application users and the developer.
3. *Implementation of the patterns for the objects*, performed by the developer.
4. *Transfer of automatically extracted entities to manual annotation mode.* The application user then does not have to spend too much time by annotating emails, he/she just adds the missed tags and deletes the superfluous ones.
5. *Manual annotation performed by the application user.* It is useful if the developer watches the process.
6. If we want to achieve high consistency, precision and recall, *several users should annotate the same set of emails.*
7. *Automatic annotation evaluation* using the precision and recall measures, or just visual evaluation in the tool by browsing the emails/files with the discovered tags (as in Figure 1).

In the subsequent annotation events and experiments this process was found to be valid and useful. In the context of Anasoft application we have identified the following objects to be extracted in the second step of the customization process:

- *Organization*: org:Name, org:RegistrationNo, org:TaxRegistrationNo
- *Person*: person:Name, person:Function
- *Contact info*: contact:Phone, contact:Email, contact:Webpage
- *Address*: address:ZIP, address:Street, address:Settlement
- *Product*: product:Name, product:Module, product:Component

[12] `http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf`

- *Document*: doc:Invoice, doc:Order, doc:Contract, doc:ChangeRequest
- *Inventory (rooms, computers, ...)*: inventory:ResID, inventory:ResType
- *Other business object*: bo:ID, bo:Type.

These objects are quite interesting and common for many enterprises, but only several of them (such as address, organization, contact or person) can be detected by the same patterns across enterprises. Of course, different patterns need to be defined for different languages. Patterns for products or business documents need to be defined for each enterprise. In addition, products are usually discovered using gazetteers. While the definition of objects and patterns can take just several hours, sometimes a bigger effort is required to create gazetteers (lists of products or services). Gazetteers need to be created manually or extracted from the company information systems if available. It is hard to estimate how much effort we have spent customizing the Anasoft Helpdesk application in the first round of experiments, since the customization process was not defined at that point in time, but in the second round (when the customization process was already defined and followed) the first step took us about 2 hours and the second step together with the seventh took us 3 hours.

In the SANET application, we have developed the Voting module. For this module, it was clear from the beginning what kind of objects needed to be detected in the emails: people, voting agreement, subject of email. Thus, not all the process steps were applied. The first and second steps were done within one hour. The third step was accomplished within 4 hours including the seventh evaluation step done only visually on about 100 emails. We found 2–3 emails that were problematic with regard to agreement detection. The voting module currently supports the agreement detection in Slovak language only.

Concerning the contact module, we have evaluated the contact detection on Spanish emails of the Fedit technology center. Basically all the needed steps (1, 3 and visually step 7) were performed by the developer, since the second step was given by the address structure: street name, number, ZIP code and city name. It took about 5 hours to create and fine-tune the address extraction patterns on 104 Spanish emails from Fedit, with 82–100 % precision and 81–94 % recall depending on the object type, which is reported in 4.2 in detail.

4.1.1 Customization Conclusion

Since the described customization process is manual (but supported by Ontea tool), we have provided this evaluation. It shows that customization for a small enterprise can be done in reasonable time. The provided tests were small so this can raise questions on scalability. In case of SANET application, we have run tests on larger email sets (more than 500 emails) and the results were satisfactory. We believe the relevance of extracted information can be improved also by allowing user to mark not discovered or wrongly discovered results directly in the email. This will be the focus of our future work.

4.2 Relevance Evaluation of Information Extraction

The point of interest in the relevance evaluation of information extraction from emails was the extraction of personal names, postal addresses and telephone numbers since such entities can be used in any enterprise, and patterns can be reused/shared among enterprises. To conduct the experiment, we have selected a subset of 50 Spanish emails in which we manually annotated these kinds of named entities in order to have a so-called golden standard for evaluation. We then evaluated the extraction results automatically by comparing them to manual annotations. We considered the information extraction result to be relevant, when it was strictly equal to the corresponding manual annotation. It means the automatic result and the manual annotation must have occupied exactly the same position in the email text. The evaluation results are summarized in the table below.

Type	Total relevant	Total extracted	Relevant extracted	Recall [%]	Precision [%]	F1 [%]
Personal name	779	788	499	64.06	63.32	63.69
Telephone number	262	178	166	63.36	93.26	75.45
Fax Number	139	127	121	87.05	95.28	90.98
Postal address	170	134	75	44.12	55.97	49.34

Table 3. Evaluation of information extraction from emails (strict match)

As we can see in Table 3, the results except for the fax number and telephone number extraction are not high. This is due to strict comparison of the extraction results against the annotated set, where many good results were rejected. For example, there were many cases where the extracted name was almost equal to the annotated one, but it differed from annotation by prefix *D.* Although this prefix could be considered as an abbreviation of the first name, in our situation it stands for Spanish *Don (Sir)*, and it was not included in the manual annotations since we did not consider the title to be a part of personal name. Postal addresses suffered from similar drawbacks: they were written in various formats, so many were extracted partially or not at all. Partially extracted addresses were classified as false matches. For example, if only a postal code with a city name was extracted from the whole address, it was a false match, although the postal code and city were located correctly. The same problem was with telephone numbers. There were many extractions, which were not identical but overlapped the manually annotated telephone numbers. This made us search for alternative ways of evaluation that would reflect these *partial successes* as well.

In the second evaluation, we considered the information extraction result to be relevant when it intersected the manual annotation. The results of evaluation were much better (Table 4) and showed us that if we enhance the information extraction from emails, we can significantly improve the recall and precision. It would also improve the performance of our social network extractor, since it operates on the results of information extraction.

Type	Total relevant	Total extracted	Relevant extracted	Recall [%]	Precision [%]	F1 [%]
Personal name	779	788	672	86.26	85.28	85.77
Telephone number	262	178	178	67.94	100.00	80.91
Fax Number	139	127	125	89.93	98.43	93.98
Postal address	170	134	133	78.24	99.25	87.50

Table 4. Evaluation of information extraction from emails (intersect match)

4.3 Social Network Evaluation

In this section we evaluate spreading activation inference algorithm on task of assigning phone numbers to persons. For both the personal names and phone numbers the social network extractor depended on the information extractor (IE). Its only responsibility was to correctly assign the received phone numbers to the received personal names. This difference in the task (compared to IE) necessitated a different evaluation methodology. Our evaluation of social network extractor focused on user needs and was inspired by the approach used to evaluate the TREC 2006 Enterprise Track in [30]. In our case, the user needs are defined as the ability to extract attributes (phone numbers) and assign them correctly to primary entities (persons). It is not necessary to have all the occurrences of each attribute value or primary entity identified in the email archive. It is acceptable that some occurrences are missed as long as each unique attribute value is identified somewhere, and correctly assigned to its proper primary entity. While the extraction of telephone numbers was fairly straightforward, a number of problems resurfaced during the extraction of personal names. Spanish personal names often consisted of four components, which the writers seldom spelled out in full. They might use three components only, and sometimes even two. On top of that, they would sometimes write their names properly accentuated, but at other times they would write without accents. We therefore decided to accept as a valid variant of personal name any extracted string consisting of at least two correctly spelled name components (accents were not required). The telephone number was considered correctly assigned if any of the acceptable variants of its owners name appeared as the first or second in the list of potential "owners" sorted by score. The results are summarized in the tables below. The number of telephone numbers in Table 5 differs from that in Table 3, because Table 5 represents only the unique telephone numbers for the whole test set of 50 emails, and not all occurences of telephone numbers in emails. Moreover, the telephone numbers were re-formatted by leaving only numeric characters in them, before they were added into the social graph.

Out of 42 relevant unique phone numbers in the test corpus (and 32 phone numbers actually extracted), two occurred in the emails that did not contain any personal name (only company name was present). These two were then irrelevant for the task of pairing phones and personal names; this is summarized in Table 6. Since there were 40 relevant unique phone numbers, we expected 40 correct pairs consisting

Total relevant	Total found	Relevant found	Recall [%]	Precision [%]
42	32	29	69	90.6

Table 5. Quantitative evaluation of the extraction of telephone numbers for social network reconstruction

of a phone number and a personal name. This was for us the *Total relevant* for the pairing task. The number of assigned pairs (shown in the *Total Found* column) is now 30, since it excludes the two irrelevant phone numbers. The number of correctly assigned phones to persons is shown in the *Relevant found* column.

Total relevant	Total found	Relevant found	Recall [%]	Precision [%]
40	30	18	45	60

Table 6. Quantitative evaluation of the assignment of phones to personal names

Table 6 implies that out of 30 found pairs, 12 were wrong. Out of these, 6 were caused by the fact that the Information Extractor failed to extract the name of the person that actually "owned" the phone number. Further 3 errors were caused by the extraction of wrong (corrupted) phone numbers, as is implied by Table 5. This means that out of 21 pairs which the social network extractor could possibly pair correctly, it only failed 3 times. This would give the theoretical precision of $18/21 = 85.7\,\%$, which demonstrates the potential of spreading activation in reconstructing social networks. The immediate conclusion is that we need to focus primarily on improving the quality of the information extraction of text-based data in multilingual contexts.

4.3.1 Social Networks Summary

In contrast to [26], where we tested our prototype on a set of 28 emails written in English, we now tested it on a set of 50 emails written in Spanish, supplied by our Commius project partner, Fedit. Spanish emails, with their accented characters and differing cultural norms (e.g. varying ways of writing personal names) represented a challenge that made a number of hidden issues visible in our prototype code. Some of them were solved on the fly, but other require a substantial redesign of the information extraction strategy. The 60 % precision of the social network extraction is less than what we obtained with English emails in [26] (precision of 77 %), but there are clear possibilities for improvement, especially at the information extraction stage. Spreading activation can deal with the lower precision of the information extraction but cannot cope with low recall, i.e. the situations when something needed was not extracted. In this respect, the main opportunity for improvement lies in making the partially good results of the information extraction (summarized in Table 4) available for the social network extractor, since now only strict IE results (shown in Table 3) are really exploited for social network reconstruction.

4.4 Evaluation Experiments Conclusion

To conclude, the success rate (precision and recall) of the Ontea IE is quite high which reflects our previous evaluations on web documents [25] as well as our present evaluation in Table 4, where the F1 measure for the intersect match was above 80 % in all cases. The developer fine-tunes the patterns and gazetteers on the target email data set in a given enterprise, and so achieves good results. We have also proved that the approach can be customized for enterprise applications in a reasonable time-frame of several hours, provided the customization is conducted by the person skilled in regular expressions.

5 CONCLUSION

We have developed a solution for simple and customizable processing of email messages with focus on enterprise emails. We believe the work described can be used in many applications in enterprise or community environment, such as knowledge management, social networks, information management or enterprise interoperability.

In the article, we have discussed our experiments with the customization of the information extraction in several enterprises and organizations. In future we plan to extend the solution with an easy user interface for user interaction with extracted data, setting up the application-specific patterns for the information extraction, hint creation and evaluation. We believe that enterprises can benefit from the proposed approach from the day one of the system installation by offering simple functionality such as search for related objects, catch a contact or show information about organizations/people. Enterprises that want to benefit from their email archives more substantially need to customize the system. Our experiments have shown that it is possible to customize it in a reasonable time-frame.

In this paper we have also discussed the email social network extraction, which employs the spreading activation algorithm on the information extraction results. We have shown that reasonable relations among business objects can be discovered based on the information hidden in the email archives. Such information can be used to populate the empty enterprise databases when installing a new information system, CRM, BPM or Help Desk. We have also shown the use of email social networks on the Email Social Network Search prototype, which exploits multi-dimensional social networks for exploring various object-to-object or object-attribute relations.

Acknowledgment

This work is partially supported by projects AIIA APVV-0216-07, Commius FP7-213876, APVV DO7RP-0005-08, SMART ITMS: 26240120005, SMART II ITMS: 26240120029, VEGA 2/0184/10. We would also like to thank Anasoft, Fedit, Aitek, Softeco, Techfin and SANET for providing us with emails for testing, and for their support with the customization of the information extraction process.

REFERENCES

[1] PewInternet Report: Online Activities 2010. May 2010, http://tinyurl.com/pewOnline10.

[2] Madden, M.—Jones, S.: Networked Workers. PewInternet report, Pew Research Center; September 24, 2008, http://tinyurl.com/pewNetWrks08.

[3] HP, The Radicati Group, Inc.: Taming the Growth of Email – An ROI Analysis (White Paper), 2005, http://tinyurl.com/RadicatiEmail05.

[4] Jones, J.: Gallup: Almost All E-Mail Users Say Internet, E-Mail Have Made Lives Better. 2001, http://tinyurl.com/Gallup01.

[5] The Radicati Group, Inc.: Email Statistics Report, 2010; Editor: Sara Radicati; http://tinyurl.com/RadicatiEmail10.

[6] META Group Inc.: 80 % of Users Prefer E-Mail as Business Communication Tool. 2003, http://tinyurl.com/MetaEmail03.

[7] Whittaker, S.—Sidner, C.: Email Overload: Exploring Personal Information Management of Email. In Proceedings of ACM CHI96, pp. 276–283, 1996.

[8] Fisher, D.—Brush, A. J.—Gleave, E.—Smith, M. A.: Revisiting Whittaker & Sidner's "Email Overload" Ten Years Later. In CSCW2006, New York ACM Press 2006.

[9] Laclavík, M.—Maynard, D.: Motivating Intelligent Email in Business: An Investigation Into Current Trends for Email Processing and Communication Research. In E3C Workshop; IEEE Conference on Commerce and Enterprise Computing; DOI 10.1109/CEC.2009.47, 2009, pp. 476–482.

[10] Balzert, S.—Burkhart, T.—Kalaboukas, K.—Carpenter, M.—Laclavík, M.—Marin, C.—Mehandjiev, N.—Sonnhalter, K.—Ziemann, J.: Appendix to D2.1.2: State of the Art in Interoperability Technology, Commius project deliverable (2009), http://tinyurl.com/CommiusSoTA.

[11] McDowell, L.—Etzioni, O.—Halevy, A.—Levy, H.: Semantic Email. in-WWW '04: Proceedings of the 13th international conference on World Wide Web. New York, NY, USA: ACM, 2004, pp. 244–254.

[12] Scerri, S. et al.: Improving Email Conversation Efficiency through Semantically Enhanced Email. In: 18th International Conference on Database and Expert Systems Applications, IEEE Computer Society 2007, pp. 490–494.

[13] Klimt, B.—Yang, Y.: Introducing the Enron Corpus. CEAS, 2004, http://www.ceas.cc/papers-2004/168.pdf, http://www.cs.cmu.edu/~enron/.

[14] Carvalho, V. R.—Cohen, W. W.: On the Collective Classification of Email "Speech Acts". In: SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on research and development in information retrieval, New York, ACM 2006, pp. 345–352.

[15] Bird, C.—Gourley, A.—Devanbu, P.—Gertz, M.—Swaminathan, A.: Mining Email Social Networks. In: MSR '06: Proceedings of the 2006 International Workshop on Mining Software Repositories, ACM, New York 2006, pp. 137–143.

[16] Culotta, A.—Bekkerman, R.—McCallum, A.: Extracting Social Networks and Contact Information from Email and the Web. In: CEAS '04: Proceedings

of the First Conference on Email and Anti-Spam 2004, http://www.ceas.cc/papers-2004/176.pdf.

[17] Laclavík, M.—Šeleng, M.—Ciglan, M.—Hluchý, L.: Supporting Collaboration by Large Scale Email Analysis. In: Cracow'08 Grid Workshop: Proceedings, Academic Computer Centre CYFRONET AGH, Kraków 2009, pp. 382–387, ISBN 978-83-61433-00-2.

[18] Diehl, C. P.—Namata, G.—Getoor, L.: Relationship Identification for Social Network Discovery. In: The AAAI 2008 Workshop on Enhanced Messaging, 2008.

[19] Judge, J.—Sogrin, M.—Troussov, A.: Galaxy: IBM Ontological Network Miner. In: 1st Conference on Social Semantic Web, Volume P-113 of Lecture Notes in Informatics (LNI) series (ISSN 16175468, ISBN 9783-88579207-9), 2007.

[20] Cunningham, H.: Information Extraction, Automatic. In: Keith Brown (Editor-in-Chief), Encyclopedia of Language & Linguistics, Second Edition: Elsevier Oxford, Volume 5, pp. 665–677.

[21] Cunningham, H.—Maynard, D.—Bontcheva, K.—Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02), Philadelphia.

[22] Kiryakov, A.—Popov, B.—Terziev, I.—Manov, D.—Ognyanoff, D.: Semantic Annotation, Indexing, and Retrieval. Elsevier's Journal of Web Semantics, Vol. 2, 2005, No. 1, http://www.ontotext.com/kim/semanticannotation.html.

[23] Cimiano, P.—Ladwig, G.—Staab, S.: Gimme' the Context: Context-Driven Automatic Semantic Annotation With C-Pankow. In WWW'05: Proceedings of the 14th international conference on World Wide Web, New York, NY, USA. ACM Press. ISBN 1-59593-046-9, 2005, pp. 332–341.

[24] Etzioni, O.—Cafarella, M.—Downey, D.—Kok, S.—Popescu, A.—Shaked, T.—Soderland, S.—Weld, D.—Yates, A.: Web-Scale Information Extraction in Knowitall (Preliminary Results). In WWW'04, 2004, pp. 100–110, http://doi.acm.org/10.1145/988672.988687.

[25] Laclavík, M.—Šeleng, M.—Ciglan, M.—Hluchý, L.: Ontea: Platform for Pattern Based Automated Semantic Annotation. Computing and Informatics, Vol. 28, 2009, pp. 555–579.

[26] Kvassay, M.—Laclavík, M.—Dlugolinský, Š.: Reconstructing Social Networks from Emails. In: Pokorný, J., SnáŠel, V., Richta, K. (Eds.): DATESO'10, 2010, pp. 50–59, ISBN 978-80-7378-116-3.

[27] Mehandjiev, N.—Gouvas, P.—Marin, C.: D5.3.1 Visual Mapping Tool (initial version). Commius project deliverable, www.commius.eu, 2009.

[28] Marin, C. A.—Carpenter, M.—Wajid, U.—Mehandjiev, N.: Devolved Ontology in Practice for a Seamless Semantic Alignment within Dynamic Collaboration Networks of SMEs. In Computing and Informatics, Vol. 30, 2011, No. 1, pp. xxx–xxx.

[29] Burkhart, T.—Werth, D.—Loos, P.—Dorn, C.: A Flexible Approach Towards Self-Adapting Process Recommendations. In Computing and Informatics, Vol. 30, 2011, No. 1, pp. xxx–xxx.

[30] Soboroff, I.—de Vries, A. P.—Craswell, N.: Overview of the TREC 2006 Enterprise Track. URL: http://tinyurl.com/TrecEnt06.

[31] Laclavík, M.—Šeleng, M.—Dlugolinský, Š.—Gatial, E.—Hluchý, L.: Tools for Email based Recommendation in Enterprise. In CENTERIS – Conference on ENTERprise Information Systems, Berlin: Springer, 2010, part I, p. 209–218. ISBN 978-3-642-16401-9. ISSN 1865-0929.

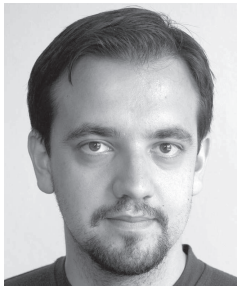
Michal Laclavík is a researcher at Institute of Informatics, Slovak Academy of Sciences. In 1999 he received his MSc degree in Computer Science and Physics. He received his Ph. D. degree in Applied Informatics with focus on Knowledge Oriented Technologies in 2006. He is the author and co-author of more than 100 publications with more than 100 citations, and participates in the Pellucid, K-Wf Grid and Commius European projects, as well as in several national projects. He has strong scientific and development expertise in email analysis, information extraction and information retrieval. He also gives lectures on Information Retrieval at Slovak University of Technology. Finally, he is one of the initiators of the Commius project focusing on email based interoperability for SMEs.

Štefan Dlugolinský is a researcher at the Institute of Informatics of Slovak Academy of Sciences. In 2009, he received his M. Eng. degree in information systems at the Faculty of Informatics and Information Technology, Slovak University of Technology. He is the author and co-author of several research papers and participates in European and national research projects. His research interests include email analysis and processing, information extraction and semantic annotation.

Martin Šeleng is a researcher at Institute of Informatics, Slovak Academy of Sciences. In 1999 he obtained his M. Sc. degree in Mathematics and Computer Sciences at the Faculty of Mathematics and Physics. He worked previously at Faculty of Economic and Informatics at the Economic University as a researcher and a teacher in the field of mathematics, statistics and computer science. He has strong expertise with email related system development and information system evaluation. He has been employed at the Institute since 2006. He is the author and co-author of several scientific papers and participates in the K-Wf Grid and Commius European projects and in several national projects. He teaches Information Retrieval at the Faculty of Informatics and Information Technologies. His research interests include email communication and large scale information processing.

Marcel Kvassay is a research assistant at the Institue of Informatics of Slovak Academy of Sciences. He graduated from the Faculty of Electrical Engineering of Slovak University of Technology in 1991. He spent a number of years as free-lance programmer and software development methodologist, working with a variety of programming languages, software development environments and methodologies. His research interests include social networks, graph theory, semantics, intelligent and distributed technologies, machine learning, knowledge management and representation.

Emil Gatial is researcher at the Institute of Informatics, Slovak Academy of Sciences. He received his M. Sc. degree in Computer Science in 2002. He received his Ph. D. degree in Applied Informatics in 2010. He is the author and co-author of several scientific papers. He participated in several European and national research projects. He has expertise in mobile agents, ontology and information processing.



Zoltán Balogh is a researcher at the Institute of Informatics of the Slovak Academy of Sciences. In 1999 he received his M. Sc. degree in Quantitative Management and Informatics. He received his Ph. D. degree in Applied Informatics in 2007. He is the author and co-author of several scientific papers and publications. He participates in several EU ICT international research projects (Pellucid, K-Wf Grid, Commius, Secricom) as well as in many national projects (VEGA, APVV, SPVV). His research focus is in ontology-based modeling, instance-based learning, web services and cloud computing and application of these technologies in real business systems.

Ladislav Hluchý is the Director of the Institute of Informatics of the Slovak Academy of Sciences and also the Head of the Department of Parallel and Distributed Computing at the institute. He received M. Sc. and Ph. D. degrees, both in Computer Science. He is R & D Project Manager, Work-package Leader in a number of 4FP, 5FP, 6FP and 7FP projects, as well as in Slovak R & D projects (VEGA, APVV). He is a member of IEEE. He is also (co-)author of scientific books and numerous scientific papers, contributions and invited lectures at international scientific conferences and workshops. He is also a supervisor and consultant for Ph. D., master and bachelor studies.

6.4 Use of Email Social Networks for Enterprise Benefit

[LAC10] LACLAVÍK, Michal - DLUGOLINSKÝ, Štefan - KVASSAY, Marcel - HLUCHÝ, Ladislav. Use of email social networks for enterprise benefit. In WI-IAT 2010 : the 2010 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology. Eds X.J. Huang, I. King, V. Raghavan, S. Rueger. - Los Alamitos : IEEE Computer Society, 2010, p. 67-70. ISBN 978-0-7695-4191-4.

2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology

Use of Email Social Networks for Enterprise Benefit

Michal Laclavík, Štefan Dlugolinský, Marcel Kvassay, Ladislav Hluchý
Institute of Informatics, Slovak Academy of Sciences
michal.laclavik@savba.sk

Abstract

The article discusses the potential methods and benefits of the analysis of social networks hidden in the enterprise and personal email archives. A proof-of-concept prototype was developed. Social network extraction and the spreading activation algorithm are discussed and evaluated.

1. Introduction

Email communication is a source of information on personal, enterprise or community networks of an individual or an organization. Email communication analysis allows the extraction of social networks with links to people, organizations, locations, topics or time. Social Networks included in email archives are becoming increasingly valuable assets in organizations, enterprises and communities, though to date they have been little explored.

Social networks in the area of email communication have been studied to some extent. For example, communication on the Apache Web Server mailing lists and its relation to CVS activity was studied in [1]. This work also introduces the problem of identifying email users' aliases. Extracting social networks and contact information from email and the Web and combining this information is discussed in [2]. Similarly, new email clients (e.g. Postbox) or plug-in Xobni[1] try to connect email social networks with web social networks like LinkedIn or Facebook. We have also performed some experiments on the extraction of social networks from large email archives and network transformations using a semantic model [3]. Another research effort [4] exploits social networks to identify relations, and tests the proposed approaches on the Enron corpus.

We are using a similar approach to that of IBM Galaxy [5] in the Nepomuk[2] project, where the concept of multidimensional social network was introduced. In this paper we show the initial results of exploiting the email social network in order to support a better understanding of email content as well as allowing applications such as search for contact, product, service, partner or supplier within organizations or communities.

2. Information Extraction

In order to provide the social network graph hidden in the email communication, the important task is to identify objects and object properties in the text and thus to formalize the email message content and context. For object identification we use the information extraction (IE) techniques. We have developed the Ontea [6] extraction and annotation tool, which uses regular expression patterns and gazetteers[3]. These patterns and gazetteers generate key-value pairs (object type – object value). Key-value pairs are then used to build the tree and the graph of social network described in section 3. Evaluation of IE is discussed in section 4.1.

3. Email Social Network Analysis

We implemented our "social network extractor and analyzer" in Java on top of the information extraction tool Ontea and open-source graphical library JUNG.[4] The novelty of our approach is in the application of the spreading activation algorithm to the twin tasks of reconstructing the social network from emails, and then efficiently searching the social graph. The prototype implementation described below is a work in progress. The evaluation of our initial experiments is provided in section 4.

In [7] we have introduced two approaches to building multidimensional social network graphs. Here

[1] http://www.xobni.com/
[2] http://nepomuk.semanticdesktop.org/
[3] Gazetteer is simply the list of keywords (e.g. list of Spanish given names) representing an object type, which are matched against the text of emails.
[4] http://jung.sourceforge.net/

978-0-7695-4191-4/10 $26.00 © 2010 IEEE
DOI 10.1109/WI-IAT.2010.126

we only use the more advanced one where IE extracts complex objects (e.g. an address) consisting of simpler objects (e.g. a street name and a postal number), and preserves the information about their physical proximity by building a hierarchical tree that includes the nodes representing the sentences, paragraphs and blocks of the message in which they were found (Figure 1). Each object is linked to all the objects in all the messages in which it was found, and can have several parallel links to one object if it occurred in it in several different positions, which is our way of recording the strength of the bond.

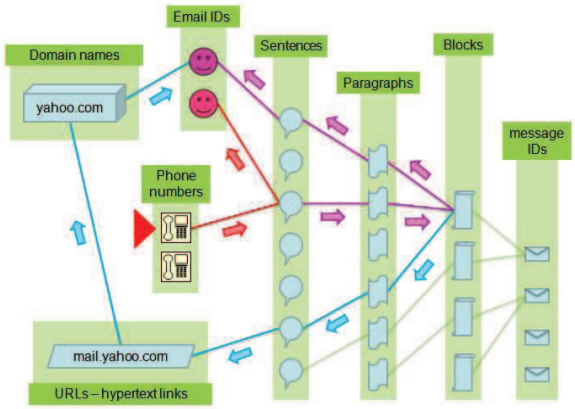


Figure 1. Spreading activation in a multipartite graph. It starts at one attribute instance (phone number) indicated by the red arrow (center left) and flows towards candidate primary entities (email IDs) by a variety of ways. Since the activation gets attenuated each time it passes through an edge, the shortest path (indicated in red) will carry over the greatest increment. But the longer paths (such as those indicated in purple and blue) can be so numerous that – depending on the value of attenuation and other parameters – their accumulated contribution may ultimately prevail

3.1. Cumulative Edge Scorer with Attenuation

In the rich tree-like structure depicted in Figure 1, it makes sense to use a more sophisticated variant of spreading activation with attenuation and activation thresholds. The structure can still be visualized as a multipartite graph (which is advantageous from the point of view of computational complexity), but the objects of each data type now require a separate partition. For the reconstruction of social network we can then use our enhanced prototype – the Cumulative Edge Scorer with Attenuation, whose algorithm is described by the following JAVA-based pseudo-code.

Breadth-first variant of Spreading Activation. `Current_wave` collection (implemented as a `HashMap` with the node as key and its `accumulated_activation` as value) contains the nodes that are going to fire. It is initialized with the attribute instance that we are trying to assign to a primary entity. In the next step, all its neighbouring nodes are placed into the `new_wave` collection as candidates for firing in the next iteration, and their accumulated activation is incremented by the value of `activation`, which reflects both the edge attenuation and the degree of the firing node. The iteration is completed when all the neighbours of all the nodes in the `current_wave` are processed. After each iteration, the number of iterations is incremented and the nodes in the `current_wave` are transferred into the `fired` collection so they do not fire again. The `current_wave` is then refilled by `new_wave.get_activated_vertices()` method that returns the nodes meeting the activation criteria. The nodes representing the primary entities never fire. They accumulate in the `new_wave` and are returned at the end of the algorithm by the `get_output_type_vertices()` method. Their accumulated activation becomes their score, and the primary entity with the highest score will "own" the attribute instance:

```
do
  for (Vertex v : current_wave)
    activation ← v.accumulated_activation();
    activation ←
      activation/(attenuation * v.degree());
    for (Edge e : v.outgoingEdges())
      Vertex w ← neighbouring_vertex(v,e);
      if (!has_fired(w))
        new_wave.smart_add(w, activation);
  total_iterations++;
  fired.add(current_wave);
  current_wave ←
       new_wave.get_activated_vertices();
  new_wave.remove_activated_vertices();
while (!done());
return  new_wave.get_output_type_vertices();
```

The resulting scores of primary entities for each attribute instance (internally stored in nested `HashMaps`) can be output in the form of a hierarchical XML. This XML can be converted to HTML by XSL transformations.

4. Evaluation

4.1. Information Extraction

We focused on the extraction of personal names and telephone numbers, which we manually annotated in each of the 50 Spanish emails to have a so-called golden standard for evaluation. We then evaluated the extraction results automatically by comparing them to manual annotations. We considered the information extraction result to be relevant, when it was strictly equal to the corresponding manual annotation. It means that both the result and the annotation must have had exactly the same position in the email text. Evaluation results are summarized in the table below.

Table 1. Evaluation of information extraction (strict match)

Type	Total relevant	Total extracted	Relevant extracted	Recall [%]	Precision [%]	F1 [%]
Person	779	788	499	64.06	63.32	63.69
Telephone	262	178	166	63.36	93.26	75.45
Fax	139	127	121	87.05	95.28	90.98

As we can see in Table 1, the results except for the fax number and telephone number extraction are not high. This is due to strict comparison of the extraction results against the annotated set during evaluation, where many good results were rejected. Similar problems appeared in personal names. Therefore we decided to perform another evaluation, with less strict matching criteria.

Table 2. Evaluation of extraction (intersect match)

Type	Total relevant	Total extracted	Relevant extracted	Recall [%]	Precision [%]	F1 [%]
Person	779	788	672	86.26	85.28	85.77
Telephone	262	178	178	67.94	100.00	80.91
Fax	139	127	125	89.93	98.43	93.98

In the second evaluation, we considered the information extraction result to be relevant when it intersected the manual annotation. The results of evaluation were much better (Table 2) and showed us that if we enhance the email information extraction, we can get better recall and precision, as well as better results in the social network extractor.

4.2. Social Network Extraction

Here the test task consisted of assigning telephone numbers to persons. For both the personal names and phones the social network extractor depended on the information extractor (IE). Its only responsibility was to correctly assign the received phone numbers to the received personal names. This difference in the task (compared to IE) necessitated a different evaluation methodology. Our evaluation of social network extractor focused on user needs and was inspired by the approach used to evaluate the TREC 2006 Enterprise Track in [8]. We defined the user needs as the ability to extract attributes (phone numbers) and assign them correctly to primary entities (persons). It is not necessary that all the occurrences of each attribute value or primary entity be identified in the email archive. It is acceptable that some occurrences are missed so long as each *unique* attribute value is identified and correctly assigned to primary entity. The telephone number was considered correctly assigned if any of the acceptable variants of its owner's name appeared as the first or second in the list of potential "owners" sorted by score. The results are summarized in the tables below.

Table 3. Quantitative evaluation of the extraction of telephone numbers for social network reconstruction

Total relevant	Total found	Relevant found	Recall [%]	Precision [%]
42	32	29	69	90.6

The number of telephone numbers in Table 3 differs from that in Table 2, because Table 3 represents only the *unique* telephone numbers for the whole set of 50 emails. Moreover, the telephone numbers were re-formatted by leaving only numeric characters in them, before they were added to the social graph.

Out of 42 relevant *unique* phone numbers in the test corpus (and 32 phone numbers actually extracted), two occurred in the emails that did not contain any personal name (only company name was present), so they were irrelevant for the task of pairing phones and personal names, which is summarized in Table 4.

Since there were 40 relevant *unique* phone numbers, we expected 40 correct pairs consisting of a phone number and a personal name. This was for us the "Total relevant" for the pairing task. The number of assigned pairs (shown in the "Total Found" column) is now 30, since it excludes the two irrelevant phone numbers. The number of correctly assigned phones to persons is shown in the "Relevant found" column.

Table 4. Quantitative evaluation of the assignment of phones to personal names

Total relevant	Total found	Relevant found	Recall [%]	Precision [%]
40	30	18	45	60

Table 4 implies that out of 30 found pairs, 12 were wrong. Out of these, 6 were caused by the fact that the Information Extractor failed to extract the name of the person that actually "owned" the phone number. Further 3 errors were caused by the extraction of wrong (corrupted) phone numbers, as implied by Table 3. This means that out of 21 pairs which the social network extractor could possibly pair correctly, it only failed 3 times. This would give the precision of 18/21 = 85.7%, which demonstrates the potential of spreading activation in reconstructing social networks. The immediate conclusion is that we need to focus primarily on improving the quality of the information extraction of text-based data in multilingual contexts.

4.3. Summary

In contrast to [7], where we tested our prototype on a set of 28 emails written in English, we now tested it on a set of 50 emails written in Spanish, supplied by our Commius project partner, Fedit. Spanish emails, with their accented characters and differing cultural norms represented a challenge. Some of them were solved on the fly, but others require a substantial redesign of the information extraction strategy. The 60% precision of the social network extraction is less than what we obtained with English emails in [7] (77% precision), but there are clear possibilities for improvement, especially at the information extraction stage, which is the base for next steps. The evaluation of the information extraction from emails is our primary focus, and we have noted several opportunities for the improvement of extraction patterns, which will then give much better results. We expect the improvement to be as good as the one shown in Table 2, since most of the errors in the social network extractor were due to the lower recall of the preceding information extraction step. Spreading activation can deal with the lower precision of the information extraction but cannot cope with low recall, i.e. the situations when something needed was not extracted. In this respect, the main opportunity for improvement lies in making the partially good results of the information extraction (summarized in Table 2) available for the social network extractor, since now only strict IE results (shown in Table 1) are really exploited for social network reconstruction.

5. Conclusion

In this paper we discussed email social network extraction based on information extraction and spread activation.

The "Social Network Extractor" component that we developed is able to process either mailboxes in mbox format or directories with email (.eml) messages, and thus extract multidimensional social network information contained in the email archive. In such a graph or network it is possible to see and exploit the links among objects such as people, time, email addresses, subjects, URLs, contact details or recipients.

The preliminary results of the extraction of social networks from email archives show that it is possible to deliver Xobni-like functionality in the enterprise or organizational context. Our approach is based on the concept of spreading activation similar to IBM Galaxy [5]. We have shown inferring relations between people and phone numbers on a limited set of emails using a simple algorithm. The success rate (precision) of the experiment was not as high as we have expected (only 60%) but on simpler English emails in [7] the precision was higher (77%). We believe there is a possibility for improvement by fine-tuning the information extraction and spreading activation algorithm. In future, we would like to infer the relations such as those between customers and services, suppliers, products and transactions, organizations and people, people and address details, and others. The approach can deliver the information needed for the adaptation of enterprise systems by filling in enterprise system databases upon installation and thus help them to offer full functionality from the beginning.

Acknowledgments

This work is partially supported by projects Commius FP7-213876, APVV DO7RP-0005-08, AIIA APVV-0216-07, VEGA 2/0184/10 and VEGA 2/0211/09. We would also like to thank Fedit for providing us with emails for testing.

10. References

[1] Bird, C., Gourley, A., Devanbu, P., Gertz, M., Swaminathan, A., "Mining Email Social Networks", In: *MSR '06: Proceedings of the 2006 Workshop on Mining Software Repositories*. ACM, New York (2006) 137–143.

[2] Culotta, A., Bekkerman, R., McCallum, A.: "Extracting Social Networks and Contact Information from Email and the Web". In: *CEAS'04*. http://www.ceas.cc/papers-2004/176.pdf

[3] Laclavík, M., Šeleng, M., Ciglan, M., Hluchý, L., "Supporting Collaboration by Large Scale Email Analysis", In: *CGW´08* (2009) 382-387. ISBN 978-83-61433-00-2

[4] Diehl, C. P., Namata, G., Getoor, L., "Relationship Identification for Social Network Discovery" In: *The AAAI 2008 Workshop on Enhanced Messaging, AAAI Conference On Artificial Intelligence*, pp 546-552 (2008)

[5] Judge, J., Sogrin, M., Troussov, A.: "Galaxy: IBM Ontological Network Miner" In: *Proceedings of the 1st Conference on Social Semantic Web*, Volume P-113 of Lecture Notes in In-formatics (LNI) series (ISSN 16175468, ISBN 9783-88579207-9). (2007)

[6] Laclavik M., Seleng M., Ciglan M., Hluchy L.: "Ontea: Platform for Pattern based Automated Semantic Annotation"; In *Computing and Informatics*, Vol. 28, 2009, 555–579

[7] Kvassay, M., Laclavík, M., Dlugolinský, Š.: "Reconstructing Social Networks from Emails". In DATESO 2010: *Proceedings of the 10th annual workshop,* pp 50-59, (2010). ISBN 978-80-7378-116-3

[8] Soboroff, I., de Vries, A.P., Craswell, N.: "Overview of the TREC 2006 Enterprise Track". (URL: http:// trec.nist.gov/pubs/trec15/papers/ENT06.OVERVIEW.pdf)

6.5 Ontea: Platform for Pattern based Automated Semantic Annotation

[LAC09] LACLAVÍK, Michal - ŠELENG, Martin - CIGLAN, Marek - HLUCHÝ, Ladislav. Ontea: Platform for pattern based automated semantic annotation. In Computing and informatics, 2009, vol. 28, no. 4, p. 555-579. (0.492 - IF2008). (2009 - Current Contents). ISSN 0232-0274.

Computing and Informatics, Vol. 28, 2009, 555–579, V 2009-Sep-16

ONTEA: PLATFORM FOR PATTERN BASED AUTOMATED SEMANTIC ANNOTATION

Michal Laclavík, Martin Šeleng, Marek Ciglan
Ladislav Hluchý

Institute of Informatics
Slovak Academy of Sciences
Dúbravská cesta 9
845 07 Bratislava, Slovakia
e-mail: michal.laclavik@savba.sk

Revised manuscript received 24 February 2009

Abstract. Automated annotation of web documents is a key challenge of the Semantic Web effort. Semantic metadata can be created manually or using automated annotation or tagging tools. Automated semantic annotation tools with best results are built on various machine learning algorithms which require training sets. Other approach is to use pattern based semantic annotation solutions built on natural language processing, information retrieval or information extraction methods. The paper presents Ontea platform for automated semantic annotation or semantic tagging. Implementation based on regular expression patterns is presented with evaluation of results. Extensible architecture for integrating pattern based approaches is presented. Most of existing semi-automatic annotation solutions can not prove it real usage on large scale data such as web or email communication, but semantic web can be exploited only when computer understandable metadata will reach critical mass. Thus we also present approach to large scale pattern based annotation.

Keywords: Semantics, semantic annotation, patterns, large scale processing

Mathematics Subject Classification 2000: 68T30, 68T10, 68Nxx

1 INTRODUCTION

Web, enterprise documents or email communication is understandable only for humans. Semantic web [1] and semantic annotation effort try to change this situation. Automated annotation tools can provide semantic metadata for semantic web as well as for knowledge management [2] or other enterprise applications [25]. Annotation can also provide semantic meta-data for other application such as e-Science [4], recommendation [38], personalization [39] or navigation [5]. Annotation solutions usually focus on the following main tasks:

- to provide annotation infrastructure and protocols
- to provide manual annotation or visualization tools
- to provide automated methods for annotation.

Annotation approaches can overlap these tasks since in every approach we need to:

- relate annotation or tags with original document
- provide customization or visualization tools for annotation process and its results
- support automation of annotation process.

Usually, whenever dealing with semantic annotation or tagging, we need to cover all of the aspects to some extent. Different strategy for annotation depends on the use of the annotation. There are a number of annotation tools and approaches such as CREAM [6] or Magpie [7] which follow the idea to provide users with useful visual tools for manual annotation, web page navigation, reading semantic tags and browsing [8] or to provide infrastructure and protocols for manual stamping documents with semantic tags such as Annotea[1], Rubby[2] or RDF annotation[3]. Even we always have to deal with infrastructure, protocols or some customization and visualization tools, in this paper we focus on automated annotation approaches for semantic metadata search and creation.

1.1 Automated Annotation State of the Art

Information Extraction – IE [9] is close to automated semantic annotation or tagging by Named Entity recognition – NE defined by series of Message Understanding Conferences (MUC). Semi-automatic annotation approaches can be divided into two groups with regard to produced results [9]:

- identification of concept instances[4] from the ontology in the text
- automatic population of ontologies with instances in the text.

[1] `http://www.w3.org/2001/Annotea/`
[2] `http://www.w3.org/TR/ruby/`
[3] `http://ilrt.org/discovery/2001/04/annotations/`
[4] In this paper instances and individuals are used with the same meaning

The Ontea method presented in the paper can provide results for both semantic annotation tasks. Semi-automatic solutions focus on creating semantic metadata for further computer processing, using semantic data in knowledge management [2] or in an enterprise application. Semi-automatic approaches are based on natural language processing [10] [11], adocument structure analysis [12] or learning requiring training sets or supervision [13]. Moreover, other annotation approaches exist, e.g. KIM[5] [21] which uses information extraction based on GATE[6] [28] information extraction system, GATE with Annie[7] extenssion or pattern-based semi-automatic solutions such as PANKOW and C-PANKOW [16] using QTag[8] patterns and Google API. To our best knowledge the only semantic annotation solution operating within distributed architecture and able to process large scale data is SemTag [14]. It uses the Seeker [14] information retrieval platform to support annotation tasks. SemTag annotates web pages using Stanford TAP ontology [15]. However, SemTag is able to identify but not create new instances in the ontology. Moreover, its results as well as TAP ontology are not available on the web for a longer period of time. For details survey on semantic annotation please see [13] [2] [20]. So far, automatic or semi-automatic approaches failed to deliver usable semantic data for semantic web applications. The main problem we see is to create general purpose semantic annotation tool usable through all domains of use – similarly as it is impossible to create one ontology which describes all domains. Thus we believe much *simple*, *scalable* and *customizable* approaches need to be used for semantic annotation. These are the gaps we are trying to address by the Ontea approach presented in the paper.

1.2 Article Structure

The structure of the article is divided into 5 chapters:

Introduction: Explains the state of the art and motivation for such work.

Ontea: Describes the Ontea aproach for semantic annotation on examples, explains the architecture, added value and integration with existing tools.

Evaluation: Discusses the success rate of Ontea.

Large Scale: Explains how Ontea can be used for large scale document annotation and evaluates the performance.

Conclusion: Summarizes results of the presented work.

2 ONTEA

Ontea identifies objects, their properties or their position in the text by applying patterns on a text. The input of the method is text and patterns and the output is

[5] `http://www.ontotext.com/kim/semanticannotation.html`
[6] `http://gate.ac.uk/`
[7] `http://gate.ac.uk/ie/annie.html`
[8] `http://www.english.bham.ac.uk/staff/omason/software/qtag.html`

key-value pairs (type – individual, individual – property), which can be transformed into RDF/OWL individuals. Ontea supports both semi-automatic annotation types:

- identification of concept instances from the ontology in the text
- automatic population of ontologies with instances in the text.

The method used in Ontea [22, 23, 24] is comparable particularly with methods such as GATE, C-PANKOW, KIM, or SemTag. While these methods try to be general purpose annotation tools through the application domains, Ontea uses simpler approach and needs to be adapted on the domain by definition or customization of used patterns. Ontea works over text applicable to an application problem domain that is described by a domain ontological model and uses regular expressions to identify relations between text and a semantic model. In addition to having pattern implementation over regular expressions, created Ontea's architecture allows simply implementation or integration of other methods based on patterns such as wrappers, solutions using document structure, XPath, language patterns, e.g. C-PANKOW or GATE.

2.1 Added Value of Ontea Approach

Simplicity: Ontea can be quite easily used since it is built on regular expression patterns (regexes). Even if not each developer is able to write regexes, regexes are widely used in text processing field. Furthermore, Ontea uses key-value pair approach to encode the result of annotation, where a *key* corresponds with a class/concept/type or a property name and a *value* corresponds with an instance/individual or a property value. More details can be found in Section 2.2.

Usability for any language: While most of advanced information extraction or annotation techniques use natural language processing (NLP) methods such as sentence decomposition or part of speech tagging (POS), such techniques do not exists or libraries (such as QTag) are not available for most of European languages, while regular expression patterns can be applicable for any language. However, NLP methods can be integrated via key-value pair result transformation.

Extensible and modular architecture: allows integration with existing techniques and allows to use simple building blocks based on key-value pairs and its transformation to prepare more advanced semantic meta data instances with properties using inference techniques; see Section 2.3.

External Data Integration: Ontea allows key-value pair transformation using external data sources to enrich annotation with extra values, properties not directly extracted from text. Such extra data can be used in ontology instance creation or search. For more details see Section 2.6 on System Connectors.

Scalability: Unlike the other annotation techniques the Ontea approach is scalable to process large data sets such as web documents, enterprise data repositories or email communication archives. See Section 4 for more details.

Suitability for "Semantic Enterprise": Information overload and problems encountered with managing available digital information is present in most of nowadays enterprises. Semantic annotation can help manage [2] information and knowledge within an enterprise. Solution such as Ontea built on reusable and configurable patterns can support contextualization and better management of information, which was partially proved and tested on email communication processing and email based recommendation system Acoma [3, 25]. Ontea implementation also allows to define patterns for objects boundaries and then group and create RDF/OWL individuals with assigned literal properties. Ontea is also able to assign discovered or created object properties to individual which represents processed text or its part. In addition, Ontea is able to enrich annotations using data from other sources such as databases, intranet systems or spreadsheets connected through system connector transformers.

2.2 Example of Use

The Ontea tool can be used in three different scenarios depending on application needs:

Ontea: searching for relevant instances in local knowledge base[9] on the basis of common patterns

Ontea creation: creation of new instances of the objects found in text

Ontea creation IR: as in the preceding example but with the use of information retrieval techniques, e.g. Lucene[10] or RFTS[11], used to identify a relevance value of created instance (see Section 2.5 for more details).

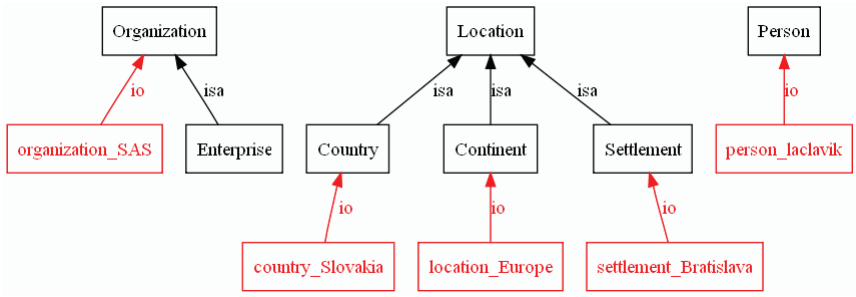


Fig. 1. Simple ontology used in examples

[9] By *knowledge base* we mean ontology repository, e.g. Sesame or Jena RDF/OWL models within this paper

[10] `http://lucene.apache.org/java/docs/index.html`

[11] `http://nazou.fiit.stuba.sk/home/?page=dairfts`

The following texts, two in English and one in Slovak, in Table 1 and ontology with several instances (Figure 1, instances are connected to their concepts via "io" arrows) illustrate the Ontea method. Moreover, the table shows examples of used patterns based on regular expressions. For first English text only one regular expression was used to find one or two words beginning with a capital letter.

#	Text	Patterns – regular expressions
1	Michal Laclavik works for Slovak Academy of Sciences located in Bratislava, the capital of Slovakia	\b([A-Z][a-z]+ +[A-Z][a-z]+|[A-Z][a-z]+)\b
2	Automobilka KIA sa rozhodla investovať pri Žiline, kde vybudovala svoju prvú továreň vEurópe. Kontakt: Kia Motors Slovakia, s.r.o. P. O. Box 2 013 01 Tepličkа nad Váhom Slovakia	*Organization* \b([\p{Lu}][-&\p{L}]+[ ]*[-&\p{L}]*[ ]*[-&\p{L}]*)[,]+s\.r\.o\. *Settlement* \b[0-9]{3}[ ]*[0-9]{2} +(\p{Lu}\p{L}+[ ]*\p{L}*[ ]*\p{L}*)[ ]*[0-9\n,]+ *Location* (v|pri) +(\p{Lu}\p{L}+)
3	Car maker KIA Motors decided to build new plant in Slovakia near the town of Zilina. It is its first plant in Europe. Contact: Kia Motors Slovakia, Ltd. P. O. Box 2 013 01 Teplicka nad Vahom Slovakia	*Location* (in|near) +([A-Z][a-z]+) *Settlement* (city|town) of ([A-Z][a-z]+ *[A-Z][a-z]+ *) *Organization* \b([A-Z][a-z]+[ ]*[A-Za-z]*[ ]*[A-Za-z]*)[,]+(Inc|Ltd)[.\s]+

Table 1. Example text

In the example #1 we have showed only finding of instances present in a knowledge base (see Figure 1 and Table 2, first row). The Slovak text example #2 and the same text in English (example #3) demonstrated not only searching for instances within a knowledge base but also their creation. The examples #2 and #3 use three patterns[12]:

- Searching for a company by *s. r. o.*, *Inc.* or *Ltd.*
- Searching for residence/domicile by ZIP code
- Searching for geographical location by means of prepositions in, near followed by a word beginning with a capital letter.

Table 2 gives also the results of examples #2 and #3. Instances created from the second row are also shown in Figure 2.

To produce results in third row of Table 2 we used lemmatization of found texts; this enables more precise creation of instances in the nominative in cases *Žilina* and

[12] Some of the patterns were simplified in Table 1 to become more readable.

Example #	Method	Annotation Results
1	Ontea	*Person* Michal Laclavik *Organization* Slovak Academy *Settlement* Bratislava *Country* Slovakia
2	Ontea create	*Location* Žiline *Location* Európe *Organization* Kia Motors Slovakia *Settlement* Teplička nad Váhom
2	Ontea create, lemmatization	*Location* Žilina *Continent* Európa *Organization* Kia Motors Slovakia *Settlement* Teplička nad Váh
3	Ontea create	*Country* Slovakia *Settlement* Zilina *Continent* Europe *Organization* Kia Motors Slovakia

Table 2. Annotation result

Európa, however, location *Teplička nad Váhom* came up wrong, as *Teplička nad Váh*. It would be appropriate to use lemmatization for several patterns only, not for all of them. Furthermore, we can direct our attention to *Európa* that is not of a *Location* but *Continent* type because the algorithm found it in a knowledge base using inference since *Continent* is a subclass of *Location*. In the creation process, the algorithm first looks into the knowledge base to find out whether instance of such type or inferred instance already exists.

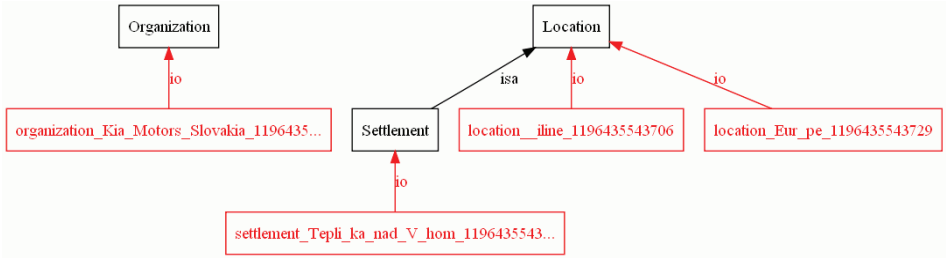


Fig. 2. Created instances in Slovak text – Ontea create

2.3 Ontea Architecture

Architecture is built to allow customizable and extensible transformation chain of key-value pairs extracted from text. The fundamental building elements of the tool are the following Java interfaces and extended and implemented objects:

ontea.core.Pattern: interface for adaptation for different pattern search. The currently implemented pattern search uses regular expressions *PatternRegExp* and *PatternXPath* to extract information from, or annotate and tag text, XML or HTML files. This interface can be used to implement more advanced patterns as well as to be used for integration of existing semantic annotation techniques.Curently we also are preparing integration with GATE.

onetea.core.Result: a class representing annotation results by means of individual of defined type/concept. Examples can be seen in Table 2. Its extensions are different types of individuals depending on implementation in ontology (Jena, Sesame) or as value and type (key-value) pairs.

ontea.transform.ResultTransformer: the interface, which after implementation, contains different types of transformations among annotation results. Thus it can transform set of results and include in transformation various scenarios of annotation such as relevance, result lemmatization, transformation of found key-value pairs (Table 2) into OWL individual in Sesame[13] or Jena[14] API implementation. It is used to transform key-value pairs into different key-value pairs represented e.g. by URI or lemmatized text value. It can be also used to eliminate irrelevant annotation results using relevance identification described in Section 2.5.

In Figure 3 you can see *Result* class, *Pattern* and*ResultTransformer* interfaces. Such design allows extending Ontea for different pattern implementations or for the integrations of existing pattern annotation solutions. It is also possible to implement various result transformations by implementing *ResultTransformer* interface, which can be used also as inputs and outputs between Map tasks in MapReduce architecture, when using Ontea algorithm on large scale data. This is discussed in Section 4.

2.4 Integration of Ontea with External Tools

The Ontea tool can be easily integrated with external tools. Some tools can be integrated by implementation of result transformers and other need to be integrated directly.

MapReduce: Large scale semantic annotation using MapReduce Architecture is described in Section 4. Integration with Hadoop requires implementation of Map and Reduce methods.

Language Identification: In order to use correct regular expressions or other patterns, we often need to identify the language of use. For this reason it is convenient to integrate Ontea with the language detection tool. We have tested Ontea with Nalit [29]. Nalit is able to identify Slovak and English texts as well as other ones, if trained.

[13] `http://openrdf.org/`
[14] `http://jena.sf.net/`

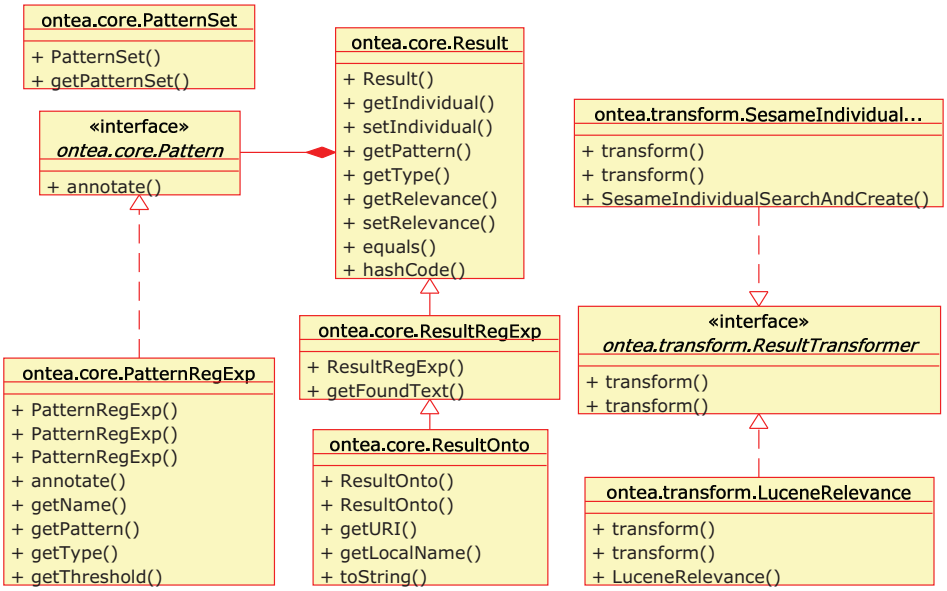


Fig. 3. Basic classes of Ontea platform

As already mentioned some integration can be done by implementing Result transformers:

Lemmatization: When concrete text is extracted as representation of an individual, we often need to lemmatize found text to find or create correct instance. For example, the capital of Slovakia can be identified in different morphological forms: *Bratislava*, *Bratislave*, *Bratislavu*, or *Bratislavou* and by lemmatization we can identify it always as individual Bratislava. We have tested Ontea with Slovak lemmatizer Morphonary [30]. It is also possible to use lemmatizers or stemmers from Snowball project [31], where Java code can be generated.

Relevance Identification: When new instance is being created or found, it is important to decide on instance relevance. This can be solved using information retrieval methods and tools. Our approach to relevance identification is described in Section 2.5.

OWL Instance Transformation: Sesame, Jena: Transformation of found key-value pairs into RDFS or OWL instances in Sesame or Jena API. With this integration, Ontea is able to find existing instances in local knowledge base if existing and to create new ones if no instance is found in knowledge base. Ontea also uses inference to found appropriate instance. For example if Ontea processes the sentence *Slovakia is in Europe* using pattern for location detection *(in|near) +(\p{Lu}\p{L}+)* the following key-value pair is detected: *Location – Europe*. If we have Location ontology with subclasses as Continents, Settlements, Coun-

tries or Cities and Europe is already present as instance of continent (see Figure 1), Ontea can detect existing *Europe* instance in the knowledge base using inference.

System Connectors Integration: Ontea allows key-value pair transformation using external data sources to enrich annotation with extra values, properties not directly extracted from text. Such extra data can be used in ontology instance creation or search. For more details see Section 2.6 on System Connectors.

2.5 Relevance Evaluation of Instances

Ontea's method of automatic annotation based on regular expressions matching showed promising results for domain specific texts. However, it suffers from frequent mismatching which leads to creation of imprecise instances of ontological concepts. We propose to overcome this obstacle by evaluating the relevance of candidate instances by the means of statistical analysis of the occurrence of the matched words in the document collection. Based on regular expression or other pattern type, Ontea identifies part of a text related to semantic context and matches the subsequent sequence of characters to create an instance of the concept.

Let us denote the sequence of words related to semantic context by C and word sequence identified as a candidate instance as I. We evaluate the relevance of the new instance by computing the ration of the close occurrence of C and I and occurrence of I:

$$\frac{close\_occurrence\,(C, I)}{occurrence\,(I)}$$

We used a prototype indexing tool – RFTS[15] that provides us with the functionality to retrieve required statistical values, computed from the whole collection of documents.

Let COLL be a collection of the documents $d_1, d_2, \ldots, d_n$: $COLL = d_1, d_2, \ldots, d_n$ Let d in COLL,$distance \in N$, and $w_1, w_2, \ldots, w_k$ are the words from natural language. Function $dist\,(d, distance, w_1, w_2, \ldots, w_k)$, where $k \leq distance$, denotes the number of distinct word sequences of the length distance containing the words $w_1, w_2, \ldots, w_k$. We compute the relevance of candidate instance as:

$$relevance\,(C, I, wordsdit) = \frac{\sum dist\,(d, wordsdist, C \cup I)}{\sum dist\,(d, (I), I)}$$

If the resulting relevance value exceeds defined threshold, the candidate word sequence I is considered to be a valid instance of the semantic concept related to sequence C. For the experimental evaluation of the approach, the threshold was set manually after inspecting the preliminary relevance values of the generated candidate instances.

[15] `http://nazou.fiit.stuba.sk/home/?page=dairfts`

Simplified relevance can also be computed using RFTS or Lucene. Percentage of occurrence of matched regular expression pattern to detected element represented by word on used document set is computed in this case as the relevance value. An example: *Google, Inc.* matched by pattern for company search: *\s+([-A-Za-z0-9][ ]*[A-Za-z0-9]*),[ ]*Inc[.\s]+*, where relevance is computed as *Google, Inc.* occurrence divided by *Google* occurrence. The idea is to omit annotation of objects which can have double meaning of the same text, e.g. *Microsoft* can be found in *Microsoft Inc.* and *Microsoft Office*. Such approach can increase the precision of annotation when needed, of course with decrease of recall. Use of RFTS or Lucene is related to *Ontea IR* scenario and *LuceneRFTS* or *LuceneRelevance* implementation of *ResultTransformer* interface. Similarly, other relevance algorithms such as cosine measure (used for example in SemTag [14]) can be implemented. Both approaches can be valuable and used with success depending on application, but gives the same results for chosen patterns within evaluation presented in Section 3.

2.6 Enriching Annotation with Information from External Systems

System connectors (SC) are middleware components providing access to legacy enterprise systems and public information systems. The purpose is to retrieve additional parameters for the input parameters specified by the software components exploiting SC. Connectors were designed to allow enriching the outputs from annotation tools and information retrieval with the data from external systems. For example, if a text annotation tool identifies the name of a person in input text (e.g. e-mail message/purchase order), the system connectors can be used to access company's partner database to retrieve the name and business ID of the company the identified person is employed in; another system connector can then be used to retrieve additional information on company from public business registry, or query billing system to retrieve last n transactions with the given company. The additional information is then processed by the annotation tool and presented to the user.

Each SC is specified by the type of resource it provides connection to (e.g. relational database, spreadsheet data resource, web application), the identification of concrete resource and metadata describing its input and output. Input metadata is a tuple (an ordered list) of parameters (name and data type). SC accepts as input tuples matching input metadata or set of key-value pairs that can be matched to input metadata. SC output is a set of tuples; the output tuples are also described by metadata (list parameter names and data types).

The metadata description of SC's input and output were introduced to allow construction of meta-connectors. Meta-connector, given a set of system connectors, is designed to match an input set of key-value pairs to the input tuple of one or several subordinate systems connectors and aggregate their outputs. The meta-connectors facilitate the integration with annotation or information retrieval systems – e.g. annotation tool configured to identify set of concepts in texts (such as e-mails, person names, company names, addresses) can identify all or only a subset of those concepts. When using meta-connector, the tool only passes the identified

values to meta-connector that automatically chooses appropriate connectors and enriches the information from available resources.

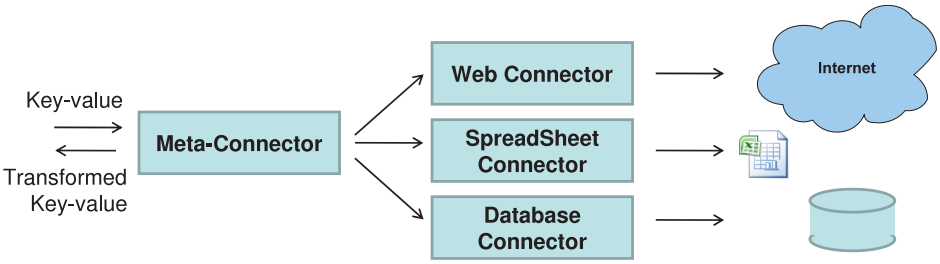


Fig. 4. System Conectors Connected to Meta-connector

To conclude, Ontea uses meta-connector to transform key-value pairs to identify instance or to gather instance properties from external systems as seen in Figure 4. For example key-value pair from Table 2 *Organization – Kia Motors Slovakia* can be transformed to *OrganizationRegistryNumber – 35876832* using web connector to Slovak Enterprise Register[16]. Instance properties can then be used when creating new ontology instance in knowledge base.

3 EVALUATION OF ONTEA ALGORITHM SUCCESS RATE

In this section we describe evaluation of Ontea annotation success rate. We also describe a test set of documents. Performance evaluation is presented in Section 4, where we also provide results on porting Ontea on MapReduce distributed architecture.

3.1 Test Set of Documents

As reference test data, we used 500 job offers downloaded from web using wrapper which prepared us some structured data. This was converted to a job offer application ontology[17], manually checked and edited according to 500 html documents representing reference job offers. Ontea processed reference html documents using the reference ontology resulting in new ontology metadata consisting of 500 job offers, which were automatically compared with reference, manually checked job offers ontology metadata.

[16] `http://www.orsr.sk/vypis.asp?ID=11731&SID=5&P=`
[17] `http://nazou.fiit.stuba.sk/home/?page=ontologies`

3.2 Target Ontological Concepts for Identification

In this test, Ontea used simple regular expressions matching from 1 to 4 words starting with a capital letter. This experiment is referred to as *Ontea*. In the second case we used domain specific regular expressions which identified locations and company names in text of job offers and Ontea also created individuals in knowledge base, while in the first case Ontea did not create extra new property individuals only searched for relevant individuals in the knowledge base. This second case is referred to as *Ontea creation*. The third case used also the previously described RFTS indexing tool or Lucene to find out if it is feasible to create a new individual using relevance techniques described earlier. This case is referred to as *Ontea creation IR*. To sum up, we conducted our experiments in 3 cases:

Ontea: searching relevant concepts in local knowledge base (KB) according to generic patterns

Ontea creation: creating new individuals of concrete application specific objects found in text

Ontea creation IR: Similar as the previous case with the feedback of RFTS or Lucene to get relevance computed using word occurrence. Individuals were created only when relevance was above the defined threshold which was set up to 10 % (see Section 2.5 for more details)

These cases were described also in Section 2.2 *Example of Use* and evaluation results for these cases are shown in Section 3.4.

We used the following regular expressions:

- Generic expression matching one or more words in text. This was used only to search concepts in the KB
  ([A-Z][-A-Za-z0-9]+[\s]+ [-a-zA-Z]+)
- Identifying geographical location in text and if not found in the KB, an individual was created
  Location: *[\s]*([A-Z][-a-zA-Z]+[ ]*[A-Za-z0-9]*)*
- Identifying a company in the text, this was used also with other abbreviations such as *Inc.* or *Ltd.*
  [\s]+([-A-Za-z0-9][ ][A-Za-z0-9]*),[ ]*Inc[.\s]+*
  or in the job offers often referenced as *Company: Company name*
  [Cc]ompany[:\s]([A-Z][-A-Za-z0-9][ ]*[A-Za-z0-9]*.*

3.3 Success Rate of the Ontea Algorithm

In this section we discuss the algorithm evaluation and success rate. To evaluate the success of annotation, we have used the standard recall, precision and F_1 measures. Recall is defined as the ratio of correct positive predictions made by the system and the total number of positive examples. Precision is defined as the ratio of correct

positive predictions made by the system and the total number of positive predictions made by the system. Recall and precision measures reflect the different aspects of annotation performance. Usually, if one of the two measures is increasing, the other will decrease. These measures were first used to evaluate techniques in Information Retrieval (IR) field by Cleverdon [17]. To obtain a better measure to describe performance, we use the F_1 measure (first introduced by van Rijsbergen [18]) which combines precision and recall measures, with equal importance, into a single parameter for optimization. F_1 measure is weighted average of the precision and recall measures.

3.4 Experimental Results of Annotation Success Rate

Ontea experimental results using precision, recall and F_1-measures are in the Table 3. In the Table 4 we show results of other semantic annotation approaches and we also list some advantages and disadvantages. The information concerning relevant annotation techniques are based on experiments presented in survey papers [13, 2], SemTag [14] and C-PANKOW [16] approaches. The success rate numbers for different tools were not evaluated on the same documents data sets and also some numbers are not available. Thus the table rows for other then Ontea tool should be taken as extra information about the method but can not be taken as direct measures to compare Ontea success rate with other methods. The column *Relevance* is similar to F_1-measures culumn in Ontea related Table 3 but in case of other methods it was evaluated by other techniques. For example for C-PANKOW, relevance is referred to as recall.

Method	Description	F_1 %	P %	R %	Disadvantages	Advantages
Ontea	regular expresions, search in knowledge base (KB)	71	64	83	high recall, lower precesion	high succes rate, generic solution, solved duplicity problem, fast algorithm
Ontea creation	regular expresions (RE), creation of individuals in KB	41	28	81	aplication specific patterns are needed low precision	support for any language
Ontea creation IR	RE, creation of individuals in KB + RFTS or Lucene relevance	62	53	79	some good results are killed by relevance identification	disambiguities are identified and not annotated, good results

Table 3. Ontea success rate evaluation (P column stands for precision, R for recall)

Method	Description	relevance %	P %	R %	Disadvantages	Advantages
SemTag	disambiguatiy check, searching in KB	high	high		works only for TAP KB	fast and generic solution
Wrapper	document structure	high	high		zero success with unknown structure	high success with known structure
PANKOW	pattern matching	59			low success rate	generic solution
C-PANKOW	POS taging and pattern matching Qtag library	74		74	suitable only for English, slow algorithm	generic solution
Hahn et al.	semantic and syntactic analysis	76			works only for English not Slovak	
Evans	clustering	41			low success rate	
Human	manual annotation	high	high	high	problem with creation of individuals duplicities, inacuracy	high recall and precesion

Table 4. Annotation tools success rate summary (P column is precision, R is recall)

In Table 3 you can see results for both semantic annotation use cases:

- identification of concept instances from the ontology in the text: row *Ontea*
- automatic population of ontologies with instances in the text: rows *Ontea creation* and *Ontea creation IR*.

The row *Ontea creation IR* case is the most important considering evaluation where we combined information retrieval and annotation techniques. By using this combination we could eliminate some not correctly annotated results. For example by using *[Cc]ompany[:\s]*([A-Z][-A-Za-z0-9][ ]*[A-Za-z0-9]** regular expression in the second case we have created and identified companies such as *This position* or *International company* which were identified as not relevant in the third case with the use of IR. Similarly *Ontea creation* identified also companies as *Microsoft* or *Oracle* which is correct and eliminated in combination with IR. This issue decreases recall and increases precision. Here it seems that IR case is not successful but the opposite is true because in many texts Microsoft is identified as products, e.g. *Microsoft Office* and if we take more text to annotate it is better not to annotate *Microsoft* as a company and decrease recall. If we would annotate *Microsoft* as a company in other

texts used in context of *Microsoft Office* we will decrease precision of annotation. It means it is very powerful to use presented annotation technique in combination with indexing in applications where high precision is required.

3.5 Success Rate Evaluation Summary

Evaluation of Ontea algorithm showed quite satisfactory results especially when identifying instances in local knowledge base – *Ontea* case. Recall was higher than 80 % and precision higher than 60 %. In *Ontea creation* case, we have achieved quite high recall (over 80 %) but precision was very low (28 %) due to the fact that patterns discovered often the same text with different meaning (similarly as in the example with *Microsoft* or *Microsoft Office* mentioned earlier). Because of such problems we have introduced IR supported instance creation where relevance whether to create instance is computed – *Ontea creation IR* case. In this case we achieved precision 53 % and recall 79 %. Such results are quite satisfactory since recall decreased only by 2 %. Ontea success rate depends much on definition of regular expression patterns and on the document collection to be annotated. We can achieve very high recall with simple patterns, but precision will be poor. When regular expression patterns will be defined carefully, precision can increase dramatically. One can argue that this method depends too much on how well the patterns are defined. This is absolutely true, but on the other hand such solution is very powerful in enterprise environment where business specific patterns need to be defined to identify products, services, invoice codes, customers or other business objects. Patterns can also be shared within a community when defined and tested for common types of objects such as people names, companies, geographical locations, addresses or contact details. In this case, one can use them directly and know expected success rates. To conclude, the state of the art semantic annotation tools declare to achieve similar success rate as Ontea. Ontea also supports both annotation tasks – population of ontologies by instance creation as well as identification of instances. However, main gaps we address by the Ontea method are *simplicity*, *customizability* and *scalability*; this would have no impact and use if algorithm success rate would be low.

4 LARGE SCALE ANNOTATION AND PERFORMANCE EVALUATION

According to our best knowledge none of the state of the art semi-automatic annotation tools except SemTag [14] provides performance evaluation results. We believe that Ontea method is one of the fastest, due to its relative simplicity – regular expression approach. For example, C-PANKOW [16] uses more advanced patterns via QTag, and also Google API for relevance calculation and thus seems to be slower. However, annotating large number of documents or periodical re-annotating of updated document collection is still time consuming when performing the computation on a single server. This is why, we have performed several experiments on parallel architectures.

We have successfully ported semantic annotation into Grid [23] with good results, but porting of application, data management and results integration was difficult and time consuming task. Thus we have also focused on different parallel and distributed architectures. In this section we discuss porting of Ontea into MapReduce architecture and its Hadoop implementation. We provide performance results on 8 nodes Hadoop cluster on email data repository.

4.1 MapReduce Architecture

MapReduce [19] architecture developed by Google was used with success on information retrieval tasks. Information extraction and pattern based annotation use similar methods such as information retrieval. This is another reason our decision to port Ontea into MapReduce architecture. We believe other well known semantic annotation or IE solutions such as C-PANKOW, KIM, GATE or various wrappers can be ported into MapReduce architecture as well, following similar approach as described in this section. Google's MapReduce [19] architecture seems to be a good choice for several reasons:

- Information processing tasks can benefit from parallel and distributed architecture with simply programming of Map and Reduce methods
- Architecture can process terabytes of data on PC clusters with handling failures
- Most information retrieval and information extraction tasks can be ported into MapReduce architecture, similar to pattern based annotation algorithms. E.g. distributed grep[18] using regular expressions, one of basic examples for MapReduce, is similar to Ontea pattern approach using regular expressions as well.
- Input and output of Map and Reduce functions are key-value pairs. Porting of Ontea as well as using Ontea transformers is thus straightforward.

In Figure 5 the main components of the MapReduce architecture are shown: Map and Reduce methods, data in distributed file system (DFS), inputs and outputs. Several data replicas are created on different nodes, when data are copied to DFS. Map tasks are executed on the nodes where data are available. Results of Map tasks are key value pairs which are reduced to results produced by Reduce method. All what a developer needs to do is to implement the Map and Reduce methods. The architecture will take care of distribution, execution of tasks as well as of fault tolerance. For more details on MapReduce see [19]. Two open source implementations of MapReduce are available:

- Hadoop [32], developed as Apache project with relation to Lucene and Nuch information retrieval systems, implemented in Java. Hadoop is well tested on

[18] Grep is a flexible search-and-replace function that can search one or more files for specified characters and/or strings

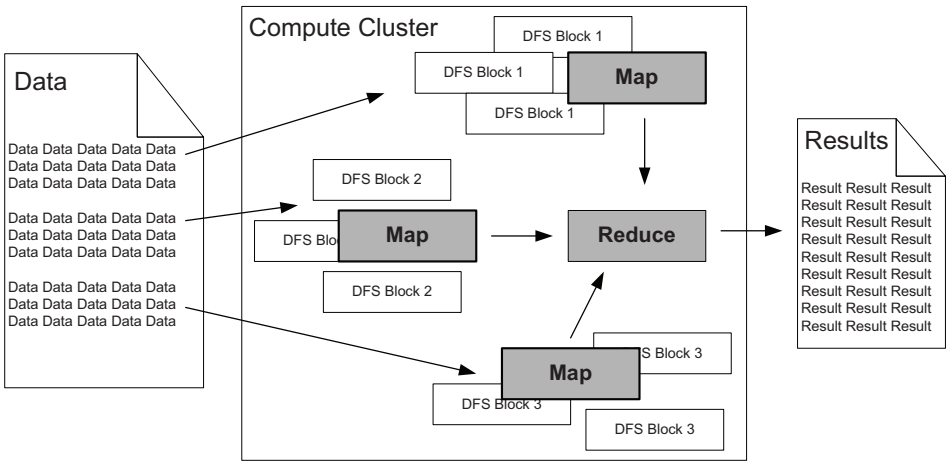


Fig. 5. MapReduce Architecture figure (source: Hadoop website)

many nodes. Yahoo! is currently running Hadoop on 10 000 nodes [33] in production environment [34].

- Phoenix [35], developed at Stanford University, implemented in C++.

4.2 Ontea Porting to Hadoop MapReduce

For porting Ontea or any semantic annotation solution it is important to understand results of annotations as well as how they can correspond to key/value pairs – input and outputs of Map and Reduce methods to be implemented in MapReduce architecture. Possible Ontea annotation results detected in test data were instances such as settlements, company names, persons or email addresses, similar as the examples shown in Table 2. In the Map method, input is a text line which is processed by Onteas regex patterns and outputs are key value pairs:

Key: string starting with detected instance type and continuing with instance value similar to Table 2 annotation results. This can be extended to return also instance properties, e.g. address, phone or email as company properties.

Value: File name with detection of instance. It can be extended with position in file, e.g. line number and character line position if needed.

The above key-value description is valid if we need to relate annotations with documents. In many cases key-value pairs in Ontea extraction and transformation are identical with key-value pairs in Ontea-Hadoop implementation, for example when extracting social network of communicating people from email communication archives [37].

Basic building blocks of Ontea are the following Java classes and interfaces described earlier, which can be extended. Here we describe them in the scope of MapReduce architecture:

ontea.core.Pattern: interface for adaptation of pattern based searching in text. The main Pattern method *Pattern.annotate()* runs inside the Map method in MapReduce implementation.

ontea.core.Result: a class which represents the result of annotation – an ontology individual. It is based on the type and value pairs similar to pairs in Table 2, *annotation results* column. Ontology results extension contains also URI of ontology individual created or found in ontology. Results are transformed into text keys as output of Map method in MapReduce implementation, or directly as key-value pairs.

ontea.transform.ResultTransformer: interface which transforms annotation results. Transformers are used in Map or Reduce methods in MapReduce implementation to transform individuals into OWL file or to eliminate some results using *Ontea IR* scenario. Transformers can be used as different Map tasks to be executed in a row depending on application scenario. For example when extracting social network from email, email addresses can be transformed into persons, organizations or else using ontology transformers above proper ontology model and knowledge base [37].

4.3 Ontea Running on Hadoop MapReduce Cluster

We wrapped up Ontea functionality into Hadoop MapReduce library. We tested it on Enron email corpus [36] containing 88 MB of data and our personal email containing 770 MB of data. We run the same annotation patterns on both email data sets, on single machine as well as on 8 node Hadoop cluster. We have used *Intel(R) Core(TM) 2 CPU* 2.40 *GHz with 2 GB RAM* hardware on all machines.

As you can see from Table 5, the performance increased 12 times on 8 nodes (16 cores) in case of large data set. In case of smaller data set it was only twice faster than on single machine and MapReduce overhead is much more visible. In Table 5 we present only 2 concrete runs on 2 different datasets, but we have executed several runs on these datasets and computational time was very similar so we can conclude that the times presented in Table 3 are very close to average. In the above tests we run only one Map method implementation and one Reduce method implementation. We would like to implement also passing Map results to another Map method as an input and thus fully exploit the potential of *ResultTransformers* in Ontea architecture. Beside the above experiments we have performed also extraction of social network and its transformation via semantic model using Ontea and Hadoop. These experiments are described in [37]. We believe we have proved scalability of Ontea by porting it on Hadoop.

Description	Enron corpus (88 MB)	Personal email (770 MB)
Time on single machine	2 min, 5 sec	3 hours, 37 mins, 4 sec
Time on 8 nodes hadoop cluster	1 min, 6 sec	18 mins, 4 sec
Performance increased	1.9 times	12 times
Launched map tasks	45	187
Launched reduce tasks	1	1
Data-local map tasks	44	186
Map input records	2 205 910	10 656 904
Map output records	23 571	37 571
Map input bytes	88 171 505	770 924 437
Map output bytes	1 257 795	1 959 363
Combine input records	23 571	37 571
Combine output records	10 214	3 511
Reduce input groups	7 445	861
Reduce input records	10 214	3 511
Reduce output records	7 445	861

Table 5. Performance and execution results

5 CONCLUSION AND FUTURE WORK

In this paper we describe briefly the state of the art in automated semantic annotation. We focus on pattern based annotation which we believe is valuable especially in enterprise applications, but can be applied also in specialized applications on the web or e-Science. We described Ontea annotation platform, which uses simple regex approach to extract key-value pairs from text and searches or creates ontology instances by various transformations of key-value pairs into ontology instances. We have described success rate of Ontea and also its modular and extensible architecture, where other pattern based annotation solution can be integrated and methods can be easily customized or extended. We provided also examples of Ontea integration with variety of tools from information retrieval or semantic web. Ontea solution was also tested and used (see demo[19]) on Job Offer application in the Nazou[20] project [26, 5]. It was also tested in enterprise environment on email communication data [3, 25]. In this paper we have also discussed how automatic semantic annotation solution such as Ontea can benefit from MapReduce distributed architecture to process a large collection of data. We demonstrated how Ontea pattern solution could be ported to implement basic Map and Reduce methods. Furthermore we provided performance results on single machine and Hadoop cluster and thus we have proved scalability of the solution.

In our future work we would like to prove Ontea usability and scalability on concrete application domains such as geographical location identification of web pages and large scale email processing to improve automated email management and se-

[19] http://nazou.fiit.stuba.sk/home/video/ontea/nazou-ontea.htm

[20] http://nazou.fiit.stuba.sk/home/?page=about

mantic searching. We would like to improve Ontea approach for instance searching by implementing gazetteer like approach used for example in GATE information extraction system. We also would like to continue developing Ontea as open source within the `Ontea.sourceforrge.net` [27] project addressing mainly *simplicity*, *customizability* and *scalability* of automated semantic annotation.

Acknowledgment

This work was partially supported by projects NAZOU SPVV 1025/2004, VEGA No. 2/6103/6, Commius FP7-213876, AIIA APVV-0216-07 and SEMCO-WS APVV-0391-06.

REFERENCES

[1] Berners-Lee, T.—Hendler, J.—Lassila, O. et al.: The Semantic Web. Scientific American, Vol. 284, 2001, No. 5, pp. 28–37.

[2] Uren, V. et al.: Semantic Annotation for Knowledge Management: Requirements and a Survey of the State of the Art. Journal of Web Semantics: Science, Services and Agents on the WWW, Vol. 4, 2005, No. 1, pp. 14–28.

[3] Laclavík, M.—Šeleng, M.—Hluchý, L.: Network Enterprise Interoperability and Collaboration using E-Mail Communication. Expanding the Knowledge Economy: Issues, Applications, Case Studies, IOS Press, Amsterdam, 2007, ISBN 978-1-58603-801-4.

[4] Chen, H.—Ma, J.—Wang, Y.—Wu, Z.: A Survey on Semantic E-Science Applications. In Computing and Informatics, Vol. 27, 2008, No. 1, pp. 1–156.

[5] Bieliková, M.—Matušíková, K.: Social Navigation for Semantic Web Applications Using Space Maps. In Computing and Informatics, Vol. 26, 2007, No. 3, pp. 281–299.

[6] Handschuh, S.—Staab, S.: Authoring and Annotation of Web Pages in Cream. In WWW '02, pp. 462–473, NY, USA, 2002. ACM Press. ISBN 1-58113-449-5, `http://doi.acm.org/10.1145/511446.511506`.

[7] Domingue, J.—Dzbor, M.: Magpie: Supporting Browsing and Navigation on the Semantic Web. In IUI '04, pp. 191–197, New York, NY, USA, ISBN 1-58113-815-6, ACM Press, 2004.

[8] Uren, V. et al.: Browsing for Information by Highlighting Automatically Generated Annotations: A User Study and Evaluation. In K-CAP '05, pp. 75–82, NY, USA, ACM Press, 2005, ISBN 1-59593-163-5.

[9] Cunningham, H.: Information Extraction, Automatic. In: Keith Brown (Editor-in-Chief): Encyclopedia of Language & Linguistics, Second Edition, Volume 5, pp. 665–677. Elsevier, Oxford, 2006.

[10] Madche, A.—Staab, S.: Ontology Learning for the Semantic Web. IEEE Intelligent Syst., Vol. 16, 2001, No. 2, pp. 72–79.

[11] Charniak, E.—Berland, M.: Finding Parts in Very Large Corpora. In Proceedings of the 37th Annual Meeting of the ACL, 1999, pp. 57–64.

[12] Glover, E.—Tsioutsiouliklis, K.—Lawrence S.—Pennock, D.—Flake, G.: Using Web Structure for Classifying and Describing Web Pages. In Proc. of the 11th WWW Conference, pp. 562–569, ACM Press, 2002.

[13] Reeve, L.—Hyoil Han: Survey of Semantic Annotation Platforms. In SAC '05, pp. 1634–1638, ACM Press, NY, USA, 2005, ISBN 1-58113-964-0.

[14] Dill, S.—Eiron, N. et al.: A Case for Automated Large-Scale Semantic Annotation. Journal of Web Semantics, 2003.

[15] Guha, R.—McCool, R.: Tap: Towards a Web of Data. http://tap.stanford.edu/.

[16] Cimiano, P.—Ladwig, G.—Staab, S.: Gimme' the Context: Context-Driven Automatic Semantic Annotation With C-Pankow. In WWW '05: Proceedings of the 14th International Conference on World Wide Web, New York, NY, USA. ACM Press, 2005, ISBN 1-59593-046-9, pp. 332–341.

[17] Cleverdon, C. W.—Mills, J.—Keen, E. M.: Factors Determining the Performance of Indexing Systems. Vol. 1–2. Cranfield, U.K., College of Aeronautics.

[18] van Rijsbergen, C. J.: Information Retrieval. Butterworth-Heinemann, Newton, MA, 1979.

[19] Dean, J.—Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. Google, Inc., OSDI '04, San Francisco, CA, December 2004.

[20] Corcho, O.: Ontology-Based Document Annotation: Trends and Open Research Problems. International Journal of Metadata, Semantics and Ontologies Vol. 1, 2006, No. 1, pp. 47–57.

[21] Kiryakov, A.—Popov, B.—Terziev, I.—Manov, D.—Ognyanoff, D.: Semantic Annotation, Indexing, and Retrieval. Elseviers Journal of Web Semantics, Vol. 2, 2005, No. 1, http://www.ontotext.com/kim/semanticannotation.html.

[22] Laclavík, M.—Šeleng, M.—Gatial, E.—Balogh, Z.—Hluchý L.: Ontology Based Text Annotation OnTeA. Information Modelling and Knowledge Bases XVIII, IOS Press, Amsterdam, 2007, Marie Duzi, Hannu Jaakkola, Yasushi Kiyoki, Hannu Kangassalo (Eds.), Frontiers in Artificial Intelligence and Applications, Vol. 154, ISBN 978-1-58603-710-9, ISSN 0922-6389, pp. 311–315.

[23] Laclavík, M.—Ciglan, M.—šeleng, M.—Hluchý, L.: Ontea: Empowering Automatic Semantic Annotation in Grid. In Paralell Processing and Applied Mathematics: 7th International Conference, PPAM 2007, R. Wyrzykowski, Jack Dongarra, Konrad Karczewski, Wasniewski (Eds.), Springer, Berlin, 2008, ISBN 978-3-540-68105-2, ISSN 0302-9743, pp. 302–311.

[24] Laclavík, M.— šeleng, M.—Hluchý, L.: Towards Large Scale Semantic Annotation Built on Mapreduce Architecture. In Computational Science, ICCS 2008, 8th International Conference, Marian Bubak, Geert Dick van Albada, Jack Dongarra, Peter M. A. Sloot (Eds.), Springer, Berlin, 2008, ISSN 0302-9743, Lecture Notes in Computer Science, Vol. 5102, Part III, pp. 331–338.

[25] Laclavík, M.—Ciglan, M.—šeleng, M.—Gatial, E.—Hluchý, L.: Future Email Services and Applications. In CEUR-WS, Proceedings of the poster and demon-

stration paper track of the 1st Future Internet Symposium (FIS '08). Vienna, Telecon Res. Center Vienna 2008, ISSN 1613-0073, Vol. 399, pp. 33–35.

[26] Návrat, P.—Bieliková, M.—Rozijanová, V.: Methods and Tools for Acquiring and Presenting Information and Knowledge in the Web. In: Proc. International Conference on Computer Systems and Technologies, CompSysTech 2005, Varna, 2005, http://ecet.ecs.ru.acad.bg/cst05/Docs/cp/SIII/IIIB.7.pdf.

[27] Ontea: Pattern based Semantic Annotation Platform. SourceForge.net project, http://ontea.sourceforge.net/, 2008.

[28] Cunningham, H.—Maynard, D.—Bontcheva, K.—Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL '02), Philadelphia, 2002.

[29] Vojtek, P.—Bieliková, M.: Comparing Natural Language Identification Methods based on Markov Processes. In: Slovko, International Seminar on Computer Treatment of Slavic and East European Languages, Bratislava, 2007.

[30] Krajči, S.—Novotný, R.: Lemmatization of Slovak Words by a Tool Morphonary. In TAOPIK (2), Vydavateľstvo STU, 2007, ISBN 978-80-227-2716-7, pp. 115–118.

[31] Snowball Project, http://snowball.tartarus.org/, 2008.

[32] Lucene-hadoop Wiki, HadoopMapReduce, 2008, http://wiki.apache.org/lucene-hadoop/HadoopMapReduce.

[33] Open Source Distributed Computing: Yahoo's Hadoop Support, Developer Network blog, http://developer.yahoo.net/blog/archives/2007/07/yahoo-hadoop.html, 2007.

[34] Yahoo! Launches World's Largest Hadoop Production Application. Yahoo! Developer Network, http://developer.yahoo.com/blogs/hadoop/2008/02/yahoo-worlds-largest-production-hadoop.html, 2008.

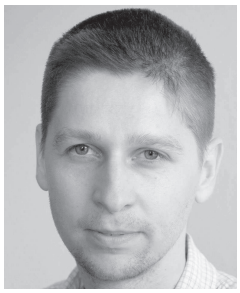
[35] The Phoenix system for MapReduce programming. http://csl.stanford.edu/christos/sw/phoenix/, 2008.

[36] Klimt, B.—Yang, Y.: Introducing the Enron Corpus. CEAS 2004, http://www.ceas.cc/papers-2004/168.pdf, http://www.cs.cmu.edu/∼enron/, 2008.

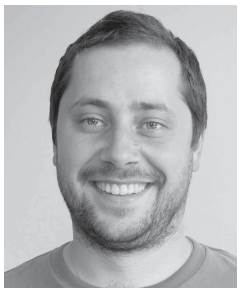
[37] Laclavík, M.—šeleng, M.—Hluchý, L.: Supporting Collaboration by Large Scale Email Analysis. CGW 2008.

[38] Andrejko, A.—Bieliková, M.: Comparing Instances of Ontological Concepts for Personalized Recommendation in Large Information Spaces. In this issue.

[39] Barla, M.—Tvarožek, M.—Bieliková, M.: Rule-based User Characteristics Acquisition from Logs with Semantics for Personalized Web-Based Systems. In this issue.

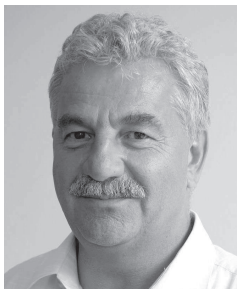


Michal Laclavík is a researcher at Institute of Informatics, Slovak Academy of Sciences. In 1999 he received his M. Sc. degree in computer science and physics. He received his Ph. D. degree in applied informatics with focus on knowledge oriented technologies in 2006. He is the author and co-author of several scientific papers, and participates in the Pellucid, K-Wf Grid and Commius European projects and in several national projects. He has strong scientific and development expertise in email based systems and semantic annotation. He also gives lectures on information retrieval at Slovak University of Technology. Finally, he is one of the initiators of the Commius project focussing on email based interoperability for SMEs. His research interests include email analysis and processing, information management and processing, and semantic annotation.



Martin Šeleng is a researcher at Institute of Informatics, Slovak Academy of Sciences. In 1999 he obtained his MSc degree in mathematics and computer science at the Faculty of Mathematics and Physics. He worked previously at Faculty of Economy and Informatics at the Economic University as a researcher and a teacher in the field of mathematics, statistics and computer science. He has strong expertise with email related system development and information system evaluation. He has been employed at the institute since 2006. He is the author and co-author of several scientific papers and participates in the K-Wf Grid and Commius European projects and in several national projects. He teaches information retrieval at the Faculty of Informatics and Information Technologies. His research interests include email communication and large scale information processing.

Marek Ciglan is a research assistant at the Institute of Informatics of Slovak Academy of Sciences. He received his M. Sc. degree in computer science in 2003. In 2008, he received his Ph. D. degree in applied informatics; with thesis focused on data replication in distributed environments. He has authored and co-authored multiple research papers and participated in several European and national research projects. His research interest is in distributed data management and distributed heterogeneous systems.



Ladislav Hluchý is the Director of the Institute of Informatics of the Slovak Academy of Sciences and also the Head of the Department of Parallel and Distributed Computing at the Institute. He received his MSc and PhD degrees, both in computer science. He is R & D Project Manager, Work-package Leader in a number of 4 FP, 5 FP, 6 FP and 7 FP projects, as well as in Slovak R & D projects (VEGA, APVT, SPVV). He is a member of IEEE, ERCIM, SRCIM, and EuroMicro consortiums, the Editor-in-Chief of the journal Computing and Informatics. He is also (co-)author of scientific books and numerous scientific papers, contributions and invited lectures at international scientific conferences and workshops. He is also a supervisor and consultant for Ph. D., master and bachelor studies.