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.

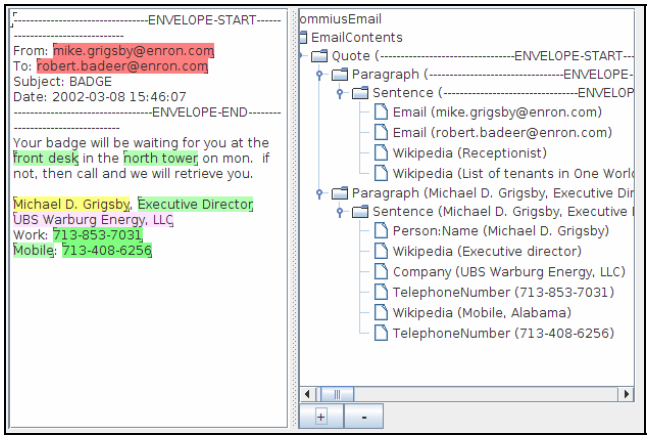

Figure 1. User interface of the IE tool Ontea [5] with highlighted extracted objects (left) and tree structure (right), which is used to build social network graph (Figure 2).

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.

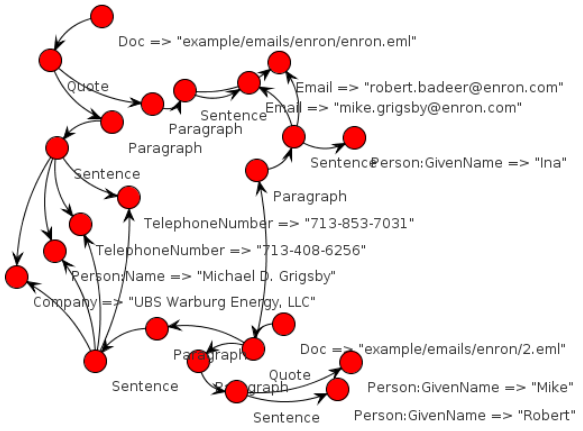

Figure 2. Social Network Graph built from two Enron emails

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.

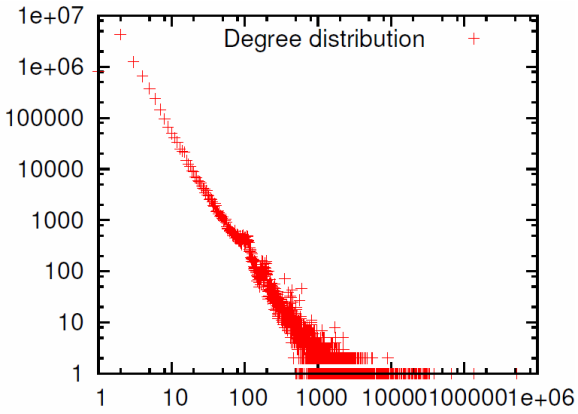

Figure 3. Enron Graph node degree distribution

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).

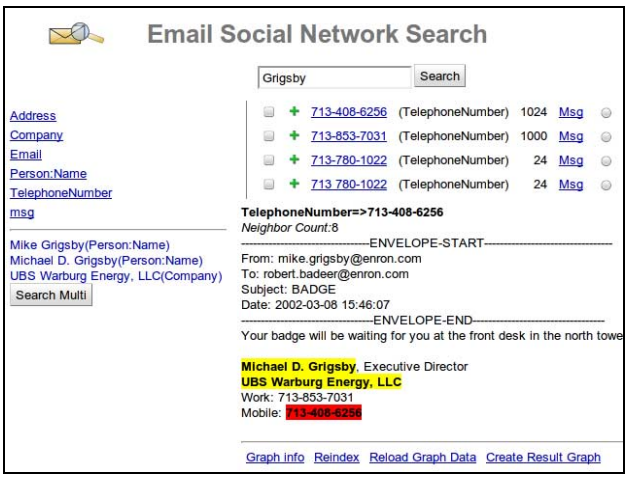

Figure 4. gSemSearch user interface.

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/