Performance Analysis of Maintaining Mobile-Based Social Network Models

PÉTER EKLER
Department of Automation and Applied Informatics
Budapest University of Technology and Economics
Magyar Tudósok Körútja 2, 1113 Budapest
HUNGARY
peter.ekler@aut.bme.hu http://www.aut.bme.hu/portal/tyrial

MÁRK ASZTALOS
Department of Automation and Applied Informatics
Budapest University of Technology and Economics
Magyar Tudósok Körútja 2, 1113 Budapest
HUNGARY
mark.asztalos@aut.bme.hu http://www.aut.bme.hu/portal/asztalosmark

Abstract: Although social networks are becoming more and more important in everyday life, their support is still not satisfactory in mobile devices. The automatic synchronization of the phone books in mobile phones with the data of the members of the social networks needs special and efficient algorithms. These algorithms cover the management of the global network model and the client software that runs on different mobile devices. In this work, we present *Phonebookmark*, a mobile-based social network implementation, which has been tested by hundreds of registered members. We present the mathematical model of the system in general that makes formal analysis possible. We introduce algorithms for the automatic discovery of similarities between contacts and members and for maintaining these similarities in the global models. We show an estimation of the complexity of the presented algorithms, which enables to extend other existing social networks by capabilities for supporting mobile devices as specified in this paper.

Key-Words: - Social networks, Modeling, Graph transformation, Complexity estimation, Mobile phones, Phonebook

1 Introduction

A social network is basically a social structure consisting of nodes that generally correspond to individuals or organizations. Nodes are connected by one or more specific types of relation. A few years ago nobody expected social networks to become so popular, and it may seem surprising that most often even senior people use social networking applications for various reasons – to find other people, send messages, manage a personal site, share photos and videos, etc. From another perspective, a social network is an environment created by people who are using it.

The increasing capabilities of mobile devices allow them to participate in social network applications. Mobile phone support in social networks is usually limited mainly to photo and video upload capabilities and access to the social network using the mobile web browser.

However, if we consider the phonebook in our mobile phone, we realize that basically it is a small part of a social network because every contact in our phonebook is related to us in a certain way. Given an implementation that allows us to upload as well as download our contacts to and from the social networking application, we can completely keep our contacts synchronized so that we can also see all of our contacts on the mobile phone as well as on the web interface. In this paper, we refer to this solution as *mobile-based social network*.

Such networks require additional resources for managing and maintaining the synchronization between the social network and the mobile devices. These large networks are usually defined by a modeling framework, which contains the definition of nodes, edge rules and allowed transformations. In this paper, we also introduce a real mobile-based social network implementation, we discuss the model and its structure, and we examine the resource requirements of such system in general. Our main contribution is that we show how design

efficient mobile-based social networks. We provide unique models for estimating the total number of contacts and similarities in the network and based on that we state the complexity of the network maintenance. The results can be used in general when a social network is being extended by mobile phone support functionalities.
The rest of the paper is organized as follows: Section 2 describes our motivation and the problem statements. Section 3 discusses related work in the field of large networks and performance evaluation. Section 4 introduces the model of mobile-based social networks. Section 5 describes the similarity handling, which is a key element of performance planning. Section 6 proposes the model for estimating the resource requirement of such solution. Finally, Section 7 concludes the paper and proposes further research area.

2 Motivation and Problem Statement

In this section first we describe the term of mobile-based social network we discuss about their advantages as motivation. Then, we introduce VMTS, which is an efficient modeling tool and we show how important it is to prepare for large number of members in the social network.
Mobile-based social networks rely on the well-known social network sites, they have a similar web user interface, but they add several major mobile phone-related functions to the system. Next we consider social networks as graphs. In case of general social networks, nodes are representing registered members and the edges between them represent the social relationships (e.g. friendship). Then we should notice that each member has a private mobile phone with a phonebook (Fig. 1). In Fig. 1, we can also see that phonebook contacts are connected to the mobile devices “owned” by different members.

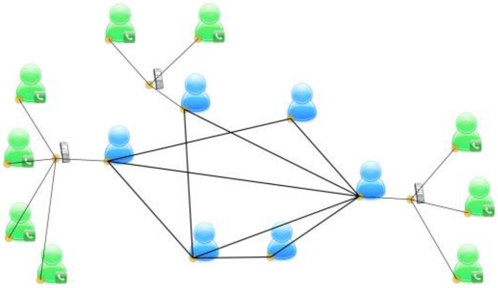


Figure 1. Phonebook-enabled social network

One of the key advantages of mobile-based social networks is that they allow real synchronization between private phonebook contacts and the social network. We need a similarity detecting algorithm in order to enable such mechanism. This algorithm is able to compare two person entries (members and private contacts, too) and determine whether they are likely similar, if so, it proposes a probability value to this detected similarity as well.
Fig. 2 represents the graph structure if the similarity detecting algorithm has finished comparing the relevant person entries.

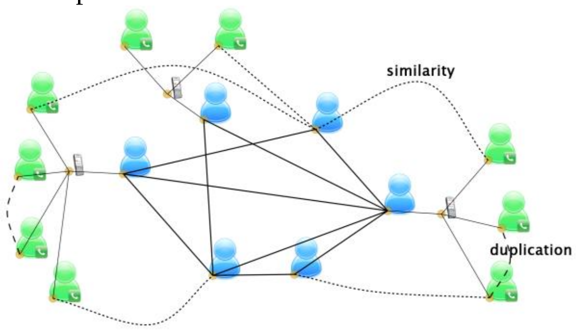


Figure 2. Detected similarities and duplications

In Fig. 2 the dotted edges between member and private contacts represent detected similarities and broken lines between two private contacts illustrate possible duplications in the phonebooks. Duplications are detected as a positive side effect of the similarity detecting algorithm.
After the similarities and duplications are detected there is a semi-automatic step, the members − who have private contacts detected as similar to other members in their phonebook − have to decide whether detected similarities are the correct ones. In addition to that, members can also decide about the correctness of detected duplications in their phonebooks. Fig. 3 represents the graph structure after some of the members have resolved the detected similarities and duplication. It can be observed that one of the private contacts of the most left member has been deleted. The other duplication link still remained on the right side because that member has not decided about it yet.

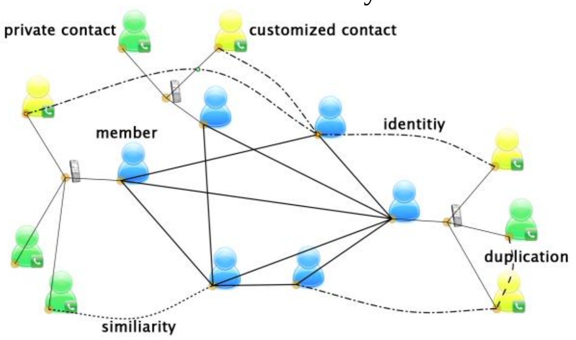


Figure 3. Resolving similarities and duplications

Moreover we can see in Fig. 3 that four of the five similarities were resolved (members found them correct) and there is still one in the system (the member has not decided yet). Resolving a similarity means that an identity link is being formed between the private contact in one's phonebook and the relevant member who represents the same person in the system. The private contacts that are linked to members via this type of identity links are called *customized contacts*. One of the key advantages of mobile-based social networks is this identity link, because if a member changes his personal detail on the web user interface (adds a new phone number, uploads a new image, changes the website address, etc), it will be automatically propagated to those phonebooks where there is a customized contact related to this member after considering privacy issues. Additional important advantages of mobile-based social networks are:

- Private contacts can be managed (list, view, edit, call, etc.) from a browser.
- Similarity detecting algorithm realizes the user if duplicate contacts are detected in its phonebook and warns about it.
- Private contacts are safely backed up in case the phone gets lost.
- Private contacts can be easily transferred to a new phone if the user replaces the old one.
- Phonebooks can be shared between multiple phones, if one happens to use more than one phone.
- It is not necessary to explicitly search for the friends in the service, because it notices if there are members similar to the private contacts in the phonebooks and warns about it.

The described mobile-based social network architecture was actually applied in the *Phonebookmark* project at Nokia Siemens Networks. Phonebookmark covered a wide range of mobile phones with the Symbian and Java ME clients. We took part in the implementation and before the public introduction it was available for a group of general users from April to December of 2008. It had 420 registered members with more than 72000 private contacts, which is a suitable number for analyzing the behavior of the network. During this period we have collected and measured different types of data related to the social network and its behavior.

Visual Modeling and Transformation System (VMTS) [15] is a metamodeling framework that makes it possible to define domain-specific modeling languages (DSMLs) [16] by means of metamodels along with complete domain specific environments (DSEs) that cover customized visualizations and editing options for the models of the domains. Moreover, VMTS supports high-level model processing techniques such as visually defined model transformations based on graph rewriting [17]. In VMTS, we have defined a DSE to model mobile-based social networks, and implemented verified model transformations for the efficient processing of the models.

We would like to highlight that the proposed mobile-based social network architecture extends the general social networks. Therefore, existing, large systems can be upgraded easier to involve mobile phones in their operation. This indicates that it is important to examine such solutions from the performance point of view.

3 Related Work

In [1] the authors have defined social network sites as web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) compile a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system. The nature and nomenclature of these connections may vary from site to site.

According to this definition, the first recognizable social network site launched in 1997. SixDegrees.com allowed users to create profiles, list their Friends and, from beginning of 1998, surf the friends lists. Later, social networks have developed rapidly and the number of features increased. Nowadays most sites support the maintenance of pre-existing social networks, others help strangers to connect people based on shared interests or activities. The sites have different ways to incorporate new information and communication tools, such as mobile connectivity, blogging, and photo/video-sharing.

As the functionality of the social networking sites were growing, the number of users increased rapidly. Handling the extending number of users efficiently is a key issue as it was visible in case of Friendster. Friendster was launched in 2002 as a social complement to Ryze. As Friendster's popularity surged, the site encountered technical and social difficulties. Friendster's servers and databases were ill-equipped to handle its rapid growth, and the site faltered regularly, frustrating users who replaced email with Friendster.

In [2], the graph structure of the Web has been investigated and it was shown that the distribution of in- and out-degree of the Web graph and the size of weekly and strongly connected components are well approximated by power law distributions.

Nazir et al. [3] showed that the in-and out-degree distribution of the interaction graph of the studied social networking applications also follow such distributions.

Distributions with an inverse polynomial tail have been first observed in 1897 by Pareto [4] (see. [5]), while describing the distribution of income in the population. In 1935 Zipf [6] and Yule [7] investigated the word frequencies in languages and based on empirical studies they derived that the frequency of the n-th frequent word is proportional to 1/n. Zipf observed similar statistical behavior in the distribution of inhabitants in cities [8].

Nowadays social networks are typical example for dynamically evolving networks. For example, Facebook is the second most popular visited websites on the Internet [9]. According to new statistics [10] Facebook has more than 400 million active users, 50% of the active users log on to Facebook in any given day, more than 35 million users update their status each day and an average user spends more than 55 minutes per day on Facebook. Facebook began in early 2004 and the above statistics show that such popular social networks can have a huge growth which has to be considered during the design of any social network site.

Mobile phones and mobile applications are another hot topic nowadays. Facebook statistics also confirm this fact, as more than 65 million active users currently accessing Facebook through their mobile devices. People that use Facebook on their mobile devices are almost 50% more active on Facebook than non-mobile users.

During the development period of Phonebookmark, we have observed other phonebook related solutions on the web. Zyb [11] and Plaxo [12] allow for synchronizing with mobile phones and managing the contacts using a web browser. Xing [13] has also mobile access, but focuses more on business relationships. There is no automatic similarity detection in these systems, thus there is no notification such as in Phonebookmark when one of a user's phonebook contacts becomes (or already is) a member of the system.

Analyzing the structure of a social network is a key issue if we consider a large number of users. The key difference between previous research and our work is that we bring together the online and "mobile" relationships based on the phonebook of the mobile devices. Therefore we have to examine the performance and resource requirements of such networks.

4 Modeling Mobile-Based Social Networks

Before we discuss the performance of mobile-based social networks, we have to examine their structure. First we describe the mathematical model of such networks, then we provide the model of it via the VMTS modeling framework.

4.1 Formal Background

Following we describe definitions, graph structure model and edge rules, related to mobile-based social networks.

Definition 1. A *mobile-based social network* is represented by the following graph:

$$\mathrm{G}_{\mathrm{MSN}} = (\mathrm{U}, \mathrm{E}), \text{ where}$$

$$\begin{aligned} U &= U_M \cup U_{Pc} \cup U_{Cc} \\ E &= E_{MM} \cup E_{MPc} \cup E_D \cup E_S \cup E_{MoCc} \cup E_{CcM} \end{aligned} \quad (1)$$

Formally the rule definition of edges is as follows:

$$E_{MM} \subseteq \{(u_M, u'_M) : u_M, u'_M \in U_M, u_M \neq u'_M\} \quad (2)$$

$$E_{MPc} \subseteq \{(u_M, u_{Pc}) : u_M \in U_M, u_{Pc} \in U_{Pc}, u'_M \in U_M, \neg\exists (u'_M, u_{Pc}) \in E_{MPc}\} \quad (3)$$

$$\mathrm{E}_{\mathrm{S}} \subseteq \{(\mathrm{u}_{\mathrm{Pc}}, \mathrm{u}_{\mathrm{M}}) : \mathrm{u}_{\mathrm{M}} \in \mathrm{U}_{\mathrm{M}}, \mathrm{u}_{\mathrm{Pc}} \in \mathrm{U}_{\mathrm{Pc}}, (\mathrm{u}_{\mathrm{Pc}}, \mathrm{u}_{\mathrm{M}}) \notin \mathrm{E}_{\mathrm{MPc}}\} \quad (4)$$

$$\mathrm{E}_{\mathrm{D}} \subseteq \{(\mathrm{u}_{\mathrm{Pc}}, \mathrm{u'}_{\mathrm{Pc}}) : \mathrm{u}_{\mathrm{Pc}}, \mathrm{u'}_{\mathrm{Pc}} \in \mathrm{U}_{\mathrm{Pc}}, \mathrm{u}_{\mathrm{Pc}} \neq \mathrm{u'}_{\mathrm{Pc}}, \mathrm{u}_{\mathrm{M}} \in \mathrm{U}_{\mathrm{M}}, \exists ((\mathrm{u}_{\mathrm{Pc}}, \mathrm{u}_{\mathrm{M}}), (\mathrm{u'}_{\mathrm{Pc}}, \mathrm{u}_{\mathrm{M}})) \in \mathrm{E}_{\mathrm{MPc}}\} \quad (5)$$

$$\mathrm{E}_{\mathrm{CcM}} \subseteq \{(\mathrm{u}_{\mathrm{Cc}}, \mathrm{u}_{\mathrm{M}}) : \mathrm{u}_{\mathrm{Cc}} \in \mathrm{U}_{\mathrm{Cc}}, \mathrm{u}_{\mathrm{M}} \in \mathrm{U}_{\mathrm{M}}, \exists \mathrm{u'}_{\mathrm{M}} \in \mathrm{U}_{\mathrm{M}}, \mathrm{u'}_{\mathrm{M}} \neq \mathrm{u}_{\mathrm{M}}, (\mathrm{u}_{\mathrm{M}}, \mathrm{u'}_{\mathrm{M}}) \in \mathrm{E}_{\mathrm{MM}}, (\mathrm{u}_{\mathrm{M}}, \mathrm{u}_{\mathrm{Cc}}) \in \mathrm{E}_{\mathrm{MoCc}}\} \quad (6)$$

$$\mathrm{E}_{\mathrm{MoCc}} \subseteq \{(\mathrm{u}_{\mathrm{Mo}}, \mathrm{u}_{\mathrm{cc}}) : \mathrm{u}_{\mathrm{Mo}} \in \mathrm{U}_{\mathrm{M}}, \mathrm{u}_{\mathrm{Cc}} \in \mathrm{U}_{\mathrm{Cc}}, \exists \mathrm{u'}_{\mathrm{M}} \in \mathrm{U}_{\mathrm{M}}, \mathrm{u'}_{\mathrm{M}} \neq \mathrm{u}_{\mathrm{Mo}}, (\mathrm{u}_{\mathrm{Mo}}, \mathrm{u'}_{\mathrm{M}}) \in \mathrm{E}_{\mathrm{MM}}, (\mathrm{u}_{\mathrm{Mo}}, \mathrm{u}_{\mathrm{Cc}}) \in \mathrm{E}_{\mathrm{CcM}}\} \quad (7)$$

Following we consider this architecture when we discuss about mobile-based social networks.

Definition 2. *Members* are a registered user of the social network, who can synchronize their phonebook with the network. We denote the set of registered members by U_M.

Members can log into the system, find and add acquaintances, upload and share personal information, write forum or blog entries, etc.

Definition 2. *Private contacts* correspond to phonebook entries of a member's mobile phone. We denote the set of private contacts in the phonebooks by U_{Pc}.

The private contacts are not shared between members; others cannot see them in the network. A private contact is saved in the system when a member synchronizes his or her phonebook with the social network.

U_M and U_{Pc} are disjoint sets. Relationships between members are represented by the edge set E_{MM} and relationships that a private contact belongs to a member are represented by the edge set E_{MPc}. The set E_D of edges indicate detected duplications in phonebooks and E_S edges indicate detected similarities between private contacts and members of the network as described previously. If a similarity was resolved between a private contact and a member the private contact is transformed to a customized contact.

Definition 3. A *customized contact* is created from a private contact when a member is similar to a private contact and the owner member of the private contact marks them as similar person. The set of customized contacts is denoted by U_{Cc}.

The set E_{CcM} of edges represent edges between customized contacts and the referred members. Finally the set E_{MoCc} of edges represent edges between owner members and their customized contacts (these will be less relevant later).

4.2 Modeling in VMTS

In the following, we present the domain-specific environment that has been implemented in VMTS for the model-based representation of mobile-based social networks. Fig. 4 presents the metamodel, which is the specification of the DSML. We define the entities of the networks, i.e. members, contacts, and phones as well as the possible relations between these entities. Note that the metamodel also contains the attribute specifications of the entities, but the presentation of all attributes is beyond the scope of this paper. However, in the following sections, we will detail the most important attributes related to the topic of this paper.

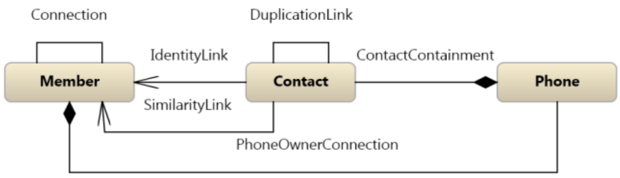


Figure 4. Mobile-based social network DSE metamodel

In VMTS, a DSE includes the metamodel and a concrete syntax extension for the instance models, which means that the models of the current metamodel have a unique domain-specific visualization along with custom functions for easier editing. We have already presented instance models as defined in VMTS; the figures in Section 2 (except from the labels that describe the types of the elements) show valid mobile-based social network models implemented in VMTS.

5 Similarity Detecting and Handling

Detecting similarities in mobile-based social networks is very important because they allow synchronization between phonebooks and the social networks when accepted. In this section, we discuss the detection of similarities and how to transform them to identity links.

5.1 Detecting similarities

In a mobile-based social network it is possible that one of our private contacts in our phonebook is similar to a member of the network, i.e. they have the same phone number or similar names, etc. In the following, we will refer to this as a *similarity*. Similarities potentially identify the same person. A similarity detection algorithm allows us to detect and accept similarities in the network and recommend possible relationships for the members. In addition to that, this algorithm enables also to recognize duplications in phonebooks.

Phonebookmark provides a semi-automatic similarity detecting and resolving mechanism. First it detects similarities and calculates a similarity weight that indicates how likely the entry identifies the same person. After the similarity has been detected, users can choose whether to accept or ignore the similarity, which is the base of the semi-automatic behavior.

During the operational period of our *Phonebookmark*, the algorithm detected 1200 similarities and users have accepted more than 90% of these (1088), which is an encouraging number for analyzing the distribution of similarities and propose a model for it. Following we refer to this rate as P_R~0.9.

5.2. Formal Background of Model Transformations in VMTS

Before presenting the model processing program that processes the similarity links in the Phonebookmark models, we introduce the basic formal background of graph rewriting-based model transformations informally based on [17]. This mathematical background makes it possible to automatically analyze and verify model transformations defined in VMTS [18].

Models are represented as attributed typed graphs, where the type graph is the formalization of the metamodel. Graphs are used because of their solid mathematical background and visual representation. The elementary modification of a graph is specified by a rewriting rule. A rule is defined by two graphs, the left-hand side (LHS) and the right-hand side (RHS), respectively. Given an input graph, its modification by a rule is performed as follows: we search for an isomorphic occurrence of LHS in the graph, and we replace it with RHS. The method what we use is referred as double-pushout (DPO) approach. We show more detailed examples for rewriting rules in Section 5.3. Note that the application of a rule contains non-determinism, since the occurrence of LHS (the match) is selected randomly. If no match can be found, the application of the rule is said to be unsuccessful. In our terminology, a *model transformation* is a model processing program that is specified by a set of rules and an additional control flow graph. The control flow graph has exactly one start node, a set of end nodes, moreover, each of the other nodes represent a rule. The control flow explicitly defines the execution order of the rules by the flow edges between the nodes. To each flow edge the value *success*, *failure*, or *dontcare* is assigned. The value *success* means that the flow edge is followed if the application of the source rule was successful, while *failure* means that the flow edge is followed if the application of the source step was unsuccessful. *Dontcare* means that the edge is followed in both cases. If there is more than one possible edge to follow after the application of the rule, one is selected non-deterministically. During the *execution* of a model transformation, we provide a concrete input model and following the control flow graph, the appropriate rules will be applied on the model under transformation, therefore, the output of the execution is always the modified input model.

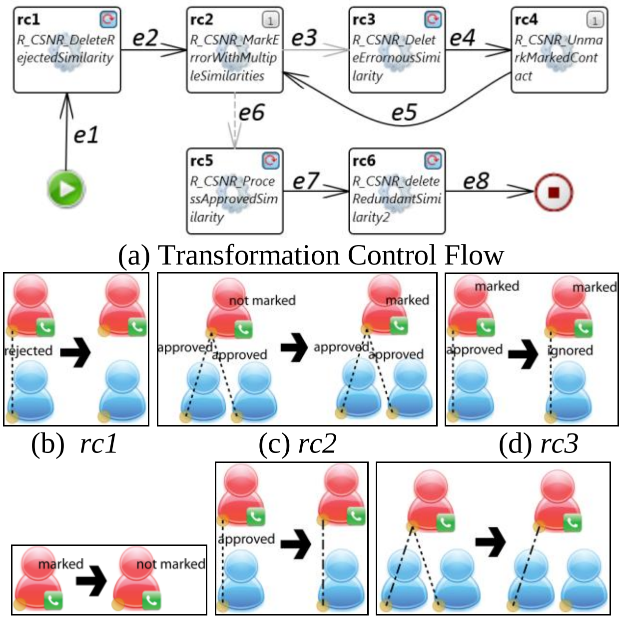
In VMTS, the graph rewriting-based transformations are defined with the use of two modeling languages: the Visual Control Flow Language (VCFL) and the Visual Transformation Definition Language (VTDL) [19]. The activity diagram-like VCFL models controls the execution order of the rewriting rules, while the rewriting rules are described with VTDL models. We show an example for a model transformation in Section 5.3.

5.3. Similarity Handling Model Transformation

When the similarity detecting algorithm detects similarity between a phonebook contact and a member of the network, a similarity link is created in the model, then, the user has to decide whether to reject it or accept this relation and mark as one that should be converted into an identity link. For this purpose, attribute *ApprovalState* has been defined for similarity links in VMTS, the value of theis attribute can be *approved*, *rejected*, or, the default value, *ignored*, which means that the user has not decided yet.

In the following, we introduce the *Similarity Handling Transformation*, which is a model transformation that processes the rejected, approved and ignored similarity links, and performs the refactoring of the input model by deleting the rejected ones and converting the approved ones into identity links. With the presented algorithm that is implemented in VMTS, we can maintain Phonebookmark models. The performance of the maintenance is detailed in Section 6. In the following, we present the formal definition of the transformation: we describe the control flow and the rules of the transformation formally. In the following, we refer to this transformation as *SHT*.

The control flow graph is defined in Fig. 5(a), we use the notation of VMTS. The dashed, gray control flow edges are followed, if the application of the source rules was unsuccessful, which happens when no matches of the left hand side can be found. The solid, gray edges are followed if the application of the previous rule was successful, while solid black edges are followed always. Rules with a circle in the top right bottom are executed exhaustively, which means that the rule is applied repeatedly, until it cannot be applied any more.



(a) Transformation Control Flow

(b) *rc1* (c) *rc2* (d) *rc3*

(e) *rc4* (f) *rc5* (g) *rc6*

Figure 5. Definition of SHT

Fig. 5 also contains the definition of the rules of the transformation. Here, we use a concrete syntax-based formal representation of each rule with specifying LHS and RHS of each of. For the sake of simplicity, attribute values are denoted as simple string in the rules. For example, LHS of rule *rc3* contains one contact, a similarity link and a member. The contact is marked, which means that the value of its *IsMarked* attribute must be true, moreover, the similarity link is approved, which means that the value of its *ApprovalState* attribute must be *approved*. Note that contacts are denoted by red icons in the rules, representing that we do not know whether they are customized. Recall that the application of rewriting rules is based on the DPO approach.

Informally, the implemented transformation works as follows: (i) Rule *rc1* removes all rejected similarity links, since it is applied exhaustively. (ii) If there is a contact that has two approved similarity links, rule *rc2* marks the contact. Marking means setting the value of the *IsMarked* attribute of the entity. (iii) *rc3* changes the approval state of an approved similarity link of a marked contact to ignored. This rule is reached only when *rc2* has been applied successfully. (iv) *rc4* removes the mark from a marked node. (v) Rule *rc5* replaces all approved similarity links with identity links. This rule is reached only when *rc2* cannot be applied. (vi) Finally, rule *rc6* removes all similarity links which come from a member that already has an identity link. Note that we assume that initially, when the transformation starts, all nodes are not marked in the input model.

6 Performance Analysis of Managing Similarities

In order to estimate the resource requirement of similarity handling, first we propose a model for estimating the number of private contacts and similarities, then, we prove that the processing of the similarities is polynomial time.

6.1 Estimating the number of private contacts

Based on the database of Phonebookmark, we were able to estimate the distribution of phonebook sizes.

Proposition 1. According to measurements, the distribution of phonebook sizes can be well approximated with an exponential distribution.

Remark. We note that the exponential distribution is a continuous distribution, but in this case, we are only interested in the values at discrete points. Our current proof is based on measurements, but the trend can be seen very well on this data set. Future work includes additional examination and measurements.

Proof. Fig. 6 shows the tail distribution of the phonebook sizes, where the *x*-axis has linear scale and the *y*-axis has logarithmic scale. The points on this figure can be well approximated with a straight line, which shows that the tail of the phonebook sizes decreases exponentially. This provides a simple empirical test for whether a random variable has an exponential distribution. The gradient of the line gives the λ parameter of the exponential distribution (Fig. 6).■

The expected value of the exponential distribution can be calculated as the reciprocal of its λ parameter, which is *0.0047*, in this measurement. Hence, the expected value of phonebook sizes according to this measurement is 212. Following we refer to $E[X_{Pc}]$ as E_{Pc}.

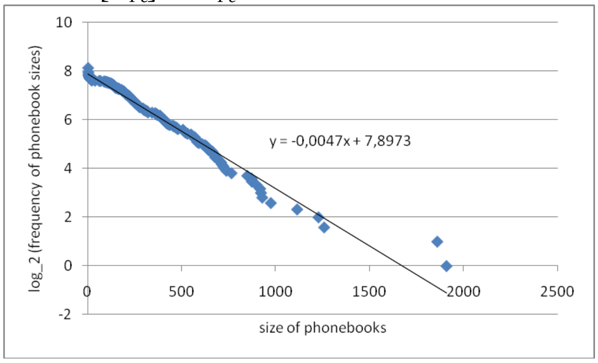


Figure 6. Size of phonebooks in Phonebookmark

Therefore, the total number of private contacts in the system is $c=E_{Pc}*m$, where m is the number of members ($m=|U_M|$).

6.2 Estimating the number of similarities

The additional resource requirements of mobile-based social networks depend mainly from the number of identity links compared to general social networks, because the number of synchronizations depends on them.

In order to estimate the resource requirement of the transformation, which resolves the accepted similarity links, first we have to provide a model for calculating the total number of similarities.

Based on the database and database logs of *Phonebookmark* we managed to measure the distribution of similarities raised by a member during registration and phonebook synchronization.

Proposition 2. In a mobile-based social network, the total number s of similarities can be estimated with the following formula: $s = m * \dfrac{\varsigma(\alpha)}{\varsigma(\alpha+1)}$,

where ζ(.) denotes the Riemann Zeta function.

Remark. The proof is based on measurements, the importance relies on that the number of similarities has not been modeled before. The trend can be seen very well on this data set.

Proof. Fig. 7 shows the complementary cumulative distribution function of the number of similarities, where the *x*-axis is the number of similarities and the *y*-axis means how many people arises at least that amount of similarities when register and synchronize.

We use logarithmically scaled *x*- and *y*-axis. We can see in Fig. 7 that the points can be well approximated with a straight line by the least squares method, thus the distribution of similarities can be well approximated by a power law. The exponent of the power law distribution is α=1.276.

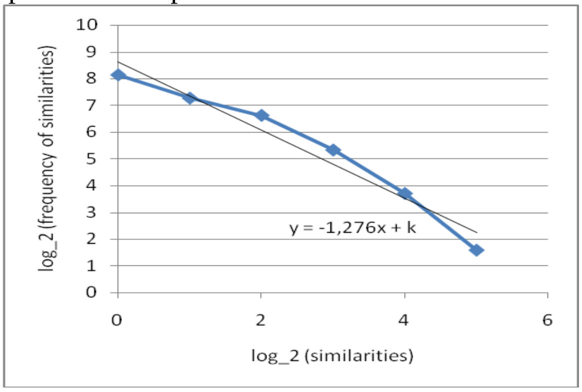


Figure 7. Distribution of similarities

According to this measurement the distribution of similarities in our case can be well approximated as follows:

$$\Pr[X \geq x] \sim x^{-1.276} \quad (8)$$

According to the results of the measurement, we model the number of similarities generated during a member registration by a probability variable *X*. More precisely, *X* models the number of similarities proposed by the automatic similarity detection algorithm.

The total number of similarities *s* in a mobile-based social network can be estimated with the following formula:

$$s = m * E[X]. \quad (9)$$

In order to calculate E[*X*], we need the probabilities Pr[*X*=*x*], which can be obtained from the complementary cumulative distribution function $\Pr[X \geq x] \sim cx^{-\alpha}$ by derivation (see, e.g. [1, 2]):

$$\Pr[X = x] \sim c' x^{-(\alpha+1)}. \quad (10)$$

In order to be a probability distribution,

$$\sum_{x=1}^{\infty} c' x^{-(\alpha+1)} = 1. \quad (11)$$

Thus, *c*'=1/ζ(α+1). Above x starts from one, because a new member registration involves at least one similarity, because the system allows registration only by invitation, therefore the new member is already in the phonebook of the inviter member.

This way the expected value is:

$$\begin{aligned} E[X] &= \sum_{x=1}^{\infty} x \Pr[X = x] \\ &= \sum_{x=1}^{\infty} x \frac{1}{\varsigma(\alpha+1)} x^{-(\alpha+1)} \\ &= \frac{1}{\varsigma(\alpha+1)} \sum_{x=1}^{\infty} x^{-\alpha} = \frac{\varsigma(\alpha)}{\varsigma(\alpha+1)}. \end{aligned} \quad (12)$$

The expected total number of similarities *s* in a mobile-based social network can be estimated with the following formula:

$$s = m * \frac{\varsigma(\alpha)}{\varsigma(\alpha+1)}. \quad (13)$$

■

For α>1, ζ(α)/ ζ(α+1) is a finite constant. The accuracy of expected value of power law distribution in case the probability variable has upper bound was shown in [14].

6.3 Complexity of the Similarity Handling Transformation

In the following, we analyze the complexity of the SHT and show that it is a polynomial time algorithm, therefore, always terminates, which enables to apply it in real-world solutions. Moreover, we show how the number of the similarity links influences the efficiency of the execution. Before starting the analysis, we provide the general statements about the complexity of a graph rewriting-based model transformation:

- Basically, the complexity depends on two components: the complexity of finding the matches in individual rules and determining how many times each rule is applied.
- The size of the input problem is the size of the input graph itself, i.e. the number of elements in the input.
- The complexity of finding a match of an LHS that contains *k* elements is at most n^k, because

in worst case, we try to match each element of the input graph to each element of LHS.

- After given a match for LHS of a rule, we assume that the modification of the input model needs a constant time.

Note that in VMTS, several optimization techniques are provided for the more efficient execution of the transformations, but these methods are not taken into account during the following analysis, because, obviously, they depend on the framework itself and here, we analyze the formal algorithm behind the graph rewriting-based model transformation SHT.

Proposition 3. The complexity of SHT (Section 5.3) is $O(c^2 \cdot s^2 \cdot n^3)$,where n denotes the number of all elements (nodes and edges) in the input mobile-based social network model, and s, c, m are the number of similarity links, contacts, and members respectively.

Remark Proposition 3 indicates that the number of private contacts and similarity edges has a great influence on the performance of the transformation, which is important, since c and s are significantly smaller than n as analyzed in the previous Section.

Proof. During the execution of the SHT, the size of the model under transformation is always at most n, since some rules delete elements, some rules only modify elements and there is only one rule (*rc5*) that creates one new element (an identity link), but it also removes one element (a similarity link). Similarly, we can show that the number of similarity links will be always at most s, and the numbers of contacts and members are not modified.

The complexity of the application of a rewriting rule r once is always polynomial, since the size the input model is finite; LHS and RHS of the rule are finite graphs, therefore, given k elements in LHS, the complexity of the application of the rule is $O(n^k + const_r) = O(n^k)$, where $const_r$ is the constant time needed to perform the modification by rule r.

The complexity of the application of rule *rc1* once is $O(c \cdot s \cdot m)$, because this is the complexity of the matching and the modification of a match needs constant time, however, *rc1* is applied exhaustively. In *rc1*, there is a similarity link that is matched in LHS, but is deleted by the rule, which results that this edge cannot be matched again by the next application. Since in the input model, we have at most s similarity links, the rule will be applied at most s times. $s \cdot O(c \cdot s \cdot n) = O(c \cdot s^2 \cdot n)$. More precisely, the rule is applied s times, but the rule is *tried* to be applied $s+1$ times, because the exhaustive application is finished when, during the last try, no matches can be found. Obviously, the matching algorithm is used in this last case as well. However, the complexity of the application of rule *rc1* is not changed by this last match, since $(s+1) \cdot O(c \cdot s \cdot n) = O(c \cdot s^2 \cdot n)$.

With similar assumptions, we can prove that the exhaustive applications of rules *rc3*, *rc5* and *rc6* are $O(c \cdot s^2 \cdot n)$, $O(c \cdot s^2 \cdot n)$, and $O(c \cdot s^2 \cdot n \cdot m^2)$ respectively. In the case of *rc6*, the maximum number of identity links is estimated by n, because we cannot use the initial number of these elements, since this number will increase during the execution of the transformation.

Finally, we analyze the single loop in our control flow, i.e. rules *rc2*, *rc3*, *rc4*. Before the first execution of *rc2*, each contact is not marked, since this is the initial requirement of the input models and rule *rc1* does not modify the *IsMarked* attribute of the contact nodes. *rc1* selects a contact that has at least two outgoing approved similarity links and marks it, hence, we have exactly one marked contact in the model after the successful application of *rc2*. If the rule is not applied successfully, the execution continues with *rc5* and never returns to *rc2*. Rule *rc3* is applied exhaustively; it modifies the approval state of all of the approved similarity links of the single marked contact. Therefore, when the execution reaches *rc4*, we still have exactly one marked contact, but it does not have any approved similarity links. Rule *rc4* removes the mark from the single marked node and continues again with *rc2*, therefore, we have no marked contact nodes, and we have the same number of contact nodes as we have before the last application of rule *rc2*. However, considering the number of contacts that have at least two approved similarity links, we can say that we have less by one than we had before the previous application of *rc2*. This means that the same contact node cannot be matched again and marked by rule *rc2*. Therefore, the loop *rc2*, *rc3*, *rc4* can be executed at most c times.

Similarly to the analysis of rule *rc1*, we can calculate the complexity of single applications of rules *rc2* and *rc4*, which are $O(c \cdot s^2 \cdot m^2)$ and $O(c)$ respectively. To summarize, the complexity of the repeated application of the loop is $(O(c \cdot s^2 \cdot m^2) + O(c \cdot s^2 \cdot n) + O(c)) \cdot c = O(c^2 \cdot (1+s^2 \cdot (m^2+n)))$.

By the previous results, we can state the sequential application of rule *rc1*, the loop, as well as rules *rc5* and *rc6* is as follows:

$O(c \cdot s^2 \cdot n) + O(c^2 \cdot (1+s^2 \cdot (m^2+n))) + O(c \cdot s^2 \cdot n) + O(c \cdot s^2 \cdot n \cdot m^2) = O(c^2+c \cdot s^2 \cdot (2n+c \cdot n^2+c \cdot m^2+n \cdot m^2))$.

Therefore, the complexity of the whole transformation ~ $O(c^2 \cdot s^2 \cdot n^3)$. ■

The results show that the transformation always terminates and is polynomial time. We emphasize how important the presence of c and s are in our result, since these values are significantly smaller than *n* as analyzed in the previous Sections.

7 Conclusion

The popularity of social networks and mobile solutions increased rapidly in the last decade. Therefore, when it is considered to extend social networks with mobile phone support or a new mobile-based social network is being created, it is very important to prepare the system for large number of members.

In this paper, we have introduced the general architecture of mobile-based social networks and we have also described *Phonebookmark* by Nokia Siemens Networks as a reference implementation. We have described the architecture of these networks with the help of the modeling tool VMTS. With the help of this model, we have described how to transform similarity links in the social network to identity links. Based on the experiences and 9 months operational period of Phonebookmark we have created a model for estimating the number of private contacts and similarities. Finally, we have proved that the similarity to identity transformation can be performed in polynomial time. Our main result is that we can estimate the complexity of the transformation for the maintenance of the network models.

Future work includes examining and comparing the data of other mobile-based social networks, extending the model for enabling social network building via VMTS and developing additional features for mobile phone support.

Acknowledgment

This work is connected to the scientific program of the "Development of quality-oriented and cooperating R+D+I strategy and functional model at BUTE" project. This project is supported by the New Hungary Development Plan (Project ID: TÁMOP-4.2.1/B-09/1/KMR-2010-0002).

References:

[1] D. M. Boyd and N. B. Ellison: Social network sites: Definition, history, and scholarship, *Journal of Computer-Mediated Communication*, Volume 13, Issue 1, 2007.

[2] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener: Graph structure in the web. In: *Proc. of the 9th int. World Wide Web conference on Computer networks*, 2000.

[3] Nazir, S. Raza and C.-N. Chuah. Unveiling Facebook: A Measurement Study of Social Network Based Applications. In: *Proc. ACM Internet Measurement Conf.*, 43-56, 2008.

[4] V. Pareto. *Course d'economie politique prof. à l'université de Lausanne*, 3 volumes, 1896-7.

[5] M. Mitzenmacher. A brief history of generative models for power law and lognormal distributions, *Internet Mathematics*, Volume 1, pages: 225-251, 2001.

[6] G. K. Zipf. The Psycho-Biology of Language. *An Introduction to Dynamic Philology*. Houghton Mifflin, Boston, MA, 1935

[7] G. U. Yule. Statistical study of literary vocabulary, *Cambridge University Press*, 1944.

[8] G. K. Zipf. Human behavior and the principle of least effort, *Addison-Wesley*, 1949.

[9] Alexa. http://www.alexa.com/topsites. June 2010.

[10] Facebook statistics, http://www.facebook.com/press/info.php?statistics, July, 2010.

[11] Zyb social networking application. http://www.zyb.com, 12th of June, 2010.

[12] Plaxo social networking application. http://www.plaxo.com, 13th of June 2010.

[13] Xing social networking application. http://www.xing.com, 15th of June 2010.

[14] P. Ekler, T. Lukovszki, The Accuracy of Similarity Calculation of Phonebook-Centric Social Networks, ICCC 2010.

[15] Visual Modeling and Transformation System website: http://vmts.aut.bme.hu/

[16] S. Kelly, J. P. Tolvanen: Domain-Specific Modeling: Enabling Full Code Generation, *Wiley-IEEE Computer Society Pr*, March 2008. ISBN 978-0470036662

[17] H. Ehrig, K. Ehrig, U. Prange, G. Taentzer: Fundamentals of Algebraic Graph Transformation, *Springer*, December 2009. ISBN 978-3642068317

[18] M. Asztalos, P. Ekler, L. Lengyel, T. Levendovszky: Applying Offline Verification of Model Transformations to Mobile Social Networks, In *Proceedings of The 4th International Workshop on Graph Based Tools (GraBaTs)*, ECEASST, September, 2010. (inpress)

[19] L. Angyal, M. Asztalos, L. Lengyel, T. Levendovszky, I. Madari, G. Mezei, T. Mészáros, L. Siroki, T. Vajk: Towards a Fast, Efficient and Customizable Domain-Specific Modeling Framework, In: *Proceedings of the IASTED international conference*. Innsbruck, Austria, 2009. pp. 11-16.