Doktorandský den ’05

Ústav informatiky
Akademie věd České republiky

Hosty – Týn nad Vltavou

5. – 10. říjen 2005

vydavatelství Matematicko-fyzikální fakulty
University Karlovy v Praze

Publikaci ”Doktorandský den ’05” sestavil a připravil k tisku František Hakl
Ústav Informatiky AV ČR, Pod Vodárenskou věží 2, 182 07 Praha 8

Všechna práva vyhrazena. Tato publikace ani žádná její část nesmí být reprodukována nebo šířena v žádné formě, elektronické nebo mechanické, včetně fotokopií, bez písemného souhlasu vydavatele.

© Ústav Informatiky AV ČR,2005
© MATFYZPRESS, vydavatelství Matematicko-fyzikální fakulty
University Karlovy v Praze 2005

ISBN *– not yet –*

Obsah

Roman Špánek: **Sharing information in a large network of users** **1**

Sharing information in a large network of users

Post-Graduate Student:
ING. ROMAN ŠPÁNEK

Department of Software Engineering, Faculty of Mechatronics,
Technical University of Liberec

roman.spanek@vslib.cz

Supervisor:
ING. JÚLIUS ŠTULLER, CSC.

Ústav informatiky AV ČR, Praha

stuller@cs.cas.cz

Field of Study:
Elektrotechnika a informatika
Classification: 2612v045

Abstract

The paper[1] describes a possible treatment of data sharing in a large network of users. The mathematical model is based on weighted hypergraphs whose nodes and edges denote the users and their relations, respectively. Its flexibility guarantees to have basic relations between users robust under frequent changes in the network connections. Approach copes with the communication/computing issues from different point of view based on a structure evolution and its further optimization in sense of keeping the *parallel* space and time complexities low. Although the idea is aimed to the field of mobile computing, it can be generalized in straightforward way to other similar environment. An experimental application is also proposed and discussed in the paper.

1. Introduction

The aim of the paper is to present a consistent model of a reconfigurable network with a distributed management useful for representing security and other relationships among users and their groups. By the *consistency* of the model we mean that the internal structure of the network must not degenerate over time to its limiting cases: small number of very large groups, or large number of small groups. In other words, our model has to have a feedback related to the fragmentation of the network into subnetworks. This we will achieve by careful use of our mathematical model, related algorithms, and implementational details.

The mobile computing area (see Figure 1) is faced by some natural limits like small bandwidth, battery power, and also by the needs of communication and computation flexibility [2],[3],[4]. While we can still expect rapid improvements in the areas of these limitations, an important question is the security of wireless networks [5]. In recent years there has been a strong progress in development of coding and cryptography which allow to have pretty secure individual transmissions. On the other hand, the security on a higher level of abstraction is still an open question and has to be addressed [6],[7],[8]. Namely, it is important to have tools for defining, evaluating, and maintaining concepts of group securities. This assumes creating an infrastructure of a network where users are restructured into smaller units (groups), and they take into account these relations in their communication [9],[10].

This paper proposes a full model which can handle security issues in mobile networks which can be dynamically evolving, or frequently reconfigured. Its mathematical basis form weighted hypergraphs. Edge

[1]The work was partially supported by the project 1ET100300419 of the Program Information Society (of the Thematic Program II of the National Research Program of the Czech Republic) “Intelligent Models, Algorithms, Methods and Tools for the Semantic Web Realisation”.

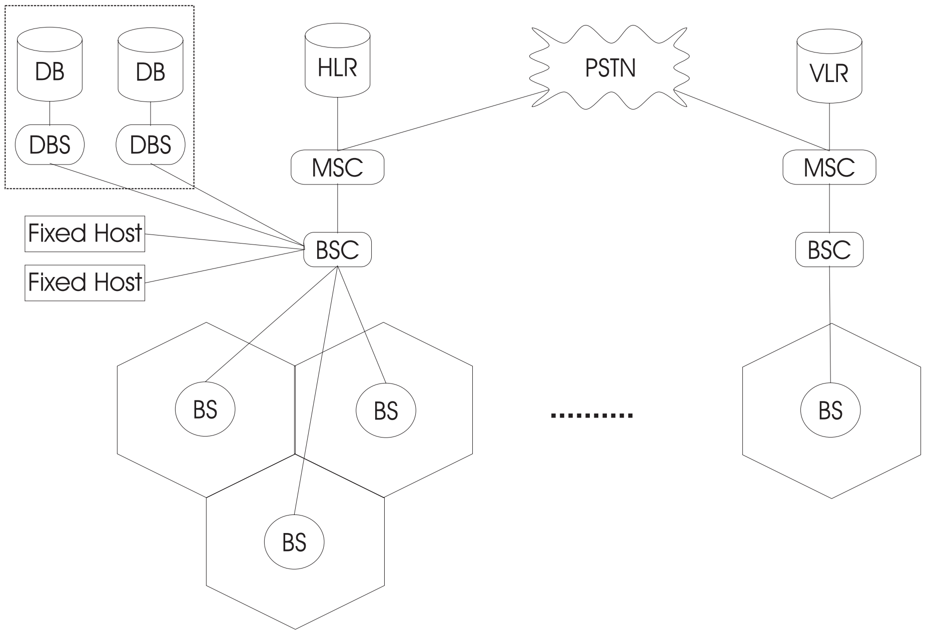

Figure 1: Mobile Database Architecture

and vertex weights can express most practical situations related to users (vertices) and relationships (hyperedges). By the term practical situations we mean not only group security issues but also possible generalizations. Network reconfigurations induce new hypergraph weights. We propose algorithms for local modification of weights. This is important for distributed runs of the algorithms. The choice of the algorithms guarantees that the model will not degenerate to a steady-state limiting case with too many or too little groups. Dynamic changes in the hypergraph model can be implemented using standard tools from numerical linear algebra [15].

The paper is organized as follows. Section 2 describes the mathematical model. Basic operations on underlying structure are proposed in Section 3. Section 4 is devoted to application of the model and the following section 5 is sharply aimed to make a brief overview on the real implementation of proposed algorithms. The paper finishes by some conclusions.

2. Mathematical Model

Our mathematical model is based on weighted hypergraphs with general real weights. Note that in many cases we will use integer weights only. In the following we will introduce some basic terminology.
The weighted hypergraph is a generalization of the concept of weighted graph which allows edges incident to more than two vertices. Formally, hypergraph $\mathcal{H}$ is a quadruple (V, E, W_V, W_E), where V is its set of vertices, $E \in 2^V$ is its edge set, and W_V and W_E present vertex and edge weights, respectively. That is, the weights are mappings from V and E to the set of reals, respectively. For simplicity, we will consider $V = \{1, \ldots, n\}$. Note that in some applications the incidence of vertices and edges is explicitly expressed by so-called vertex and edge connectors [11]. Here we prefer the classical definition [12] given above which is suitable for our purposes. Most of the important concepts from graphs can be easily generalized to hypergraphs. Here we will mention only some of them. The weights may express further properties of graphs and hypergraphs, in addition to their connections. For instance, they can express quality or priority of communication, cost issues, efficiency of evaluations in individual vertices. As we will emphasize later, in our application related to security in networks, the vertex weights will express *reliability* of users, and edge weights correspond to *security of vertex connections*.

3. Basic Operations on Hypergraphs and Related Data Structures

This section is devoted to a brief overview of basic operations used in our application, and underlying data structures. Note that application-related issues will be treated later.

A basic assumption of the mobile computing and/or communication is that the whole hypergraph model is not centralized but distributed. In addition to global network properties we need to store most of the data locally, or in close neighborhood of vertices. Small to medium grids may assume that individual vertices store: a/ vertex related information, b/ their adjacency sets c/ weights of incident (hyper)edges depending on the size of their local memories. If the local memory allows it, they may store also more levels of information. They can store vertices and edges up to their k-adjacencies for some $k >= 1$ which is what we will assume. Larger grids may need a domain-based distribution [13] for faster execution/communication even if the local memories are relatively large. Specific vertices (group leaders) contain all relevant information on weights and incidences for whole domains. In our implementation they correspond to meta-vertices, i.e. seed vertices of meta-associations related to groups of users. The following text describes some application related issues which help to reflect specific model features.

4. Application

A crucial assumption is that the application is distributed. That is, the representation and the updates must be implemented with respect to this. Therefore, all the proposed ideas have to keep the *parallel* space and time complexity low.

4.1. Group representation

Two extremal types of distributed representation of groups (subgraphs) are as follows. First, a user (note that a user is equivalent to a vertex in the application section) stores all information (connections, edge weights) related only to its adjacency set (set of all its neighbors). In this case, communication with neighbors has time complexity O(1), space complexity is small, but overall communication with distant neighbors might be expensive. Second, users can store more levels of neighbors (for the concept of levels in graphs which can be easily generalized to hypergraphs see [14]). That is information on neighbors of neighbors can be stored, and so on. Using more explicit representation of more distant vertices decreases average time complexity to communicate with other users but (local) space complexity may be rather high. Complexities are described more in detail in section 4.5. Our representation is a hierarchical with more *layers* of hierarchy. The *base layer* is formed by standard users. Each group has a representant that is responsible for group management called a *Group leader* (GL). This vertex is also responsible for communication support to other groups. The group leaders form the *enhanced layer* of users. Communication is allowed only between nodes on the same level or to one level deeper. This combines advantages of both extremal possibilities. The vertices are members of groups which dynamically evolve depending on changes in vertex and edge hypergraph weights. A group joining process is managed by a special node, called *Group Gate* (GG).

An important assumption for distribution of vertices into groups is that strong hypergraph components will be a part of the same group. That is, the connections among vertices form an *acyclic* hypergraph.

4.2. Edge evaluation

Up to now we have explained the role of vertex weights. Edge weights are the basis of the dynamic behavior of the network of users. In other words, by their appropriate evaluations and reevaluations the network thus gets its *Social Network Property*, i.e. the ability to describe social relationships in terms of a weighted hypergraph model. Consequently, in this subsection we will present a consistent set of rules and algorithms which do represent consistent static and dynamic properties of a secure network. The fact that an edge has a high weight we will equivalently call that the connected vertices (users) have a good mutual relationship.

4.2.1 Edges' evaluation rules: Here we will describe possible cases which force evaluation and reevaluation of hypergraph edges. The procedures can be applied in all layers of vertices of a hierarchical model but we will not distinguish this here. Although we have taken into consideration full spectrum of

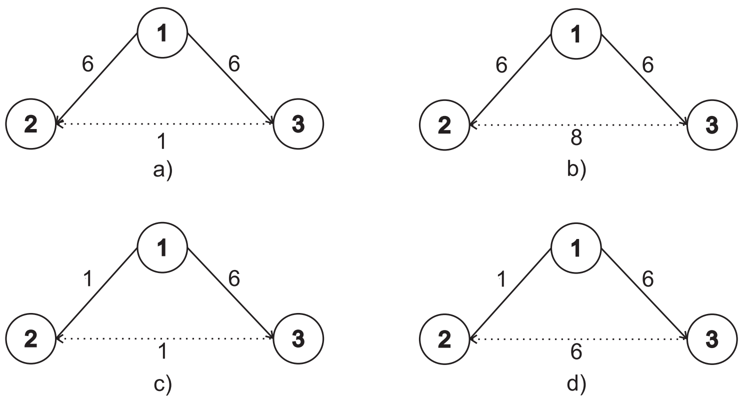

Figure 2: Four possible cases.

possible modifications in edge reevaluation and addition, one which worths mentioning here is a case depicted in Figure 2. A newly added edge is plotted in a dotted line. When the edge has been added into the structure its weight might by in imbalance with the old ones (Figure 2 d)). If so there is a need to do some reevaluation that will preserve the weights in the consistent state. A possible approach is to decrease the weight between vertices 1 and 3 to the weight of edge 1-2. Another possibility is to give to all vertices in the cycle their average value. The decision can be made automatically based *on user trigger* (see section 4.4) or chosen by involved users.

Once edge has been re-evaluated, it might cause some other re-evaluations. Consequently, this might result in huge computational overhead. In cases a), b), and c) we do insist that the further re-evaluation is not necessary. On the other hand, in d) this is not longer true. Because of imbalance in edges' weights (namely edges 1-2, 1-3), re-evaluation has to be done to preserve security properties of the structure based on mutual relationships. When the new edge has weight 6, the weights of edges (1-2, 1-3) must be re-evaluated. Once the weights were changed we are done since we get case a) or c).

4.3. Optimization and self-monitoring

We will introduce two important terms *graph condensation* and *graph expansion*. *The graph condensation* means representing groups of nodes by a single item. It frequently occurs when new nodes and/or new edges are added as shown in the previous Section 4.2.1.

Graph expansion will be performed if a group of users fulfills $|K| > \delta\,|K_{avg}|$, where $|K|$ is the component size, δ is a non-negative real number, and $|K_{avg}|$ is the average size of all groups of users in the structure. In this case, the component is too big and cannot be efficiently managed, and a graph expansion is initiated. It divides the group into a set of smaller ones. For each newly created component a GL is found and necessary data structures are set up as mentioned in section 4.2.1.

For optimization and diagnosing purposes, additional parameters are defined. The one which is worth mentioning here is a *time stamp* that stores the last moment when the edge was used. It helps to evaluate an importance of the edge. At specified time moments, the group leader runs a program structure *Trigger* (see section 4.4) to get a difference between the time stamp and the real time, given as $Time\_Diff = Time\_Stamp - Actual\_time$. If the size of $Time\_Diff$ is too small (with respect to a threshold), the edge is deleted.

4.4. Triggers

Our implementation makes use of triggers in order to evaluate and reevaluate the role of users in the network. In fact, this is a global communicating mechanism which keeps the overall consistency of our distributed model. The most important role of triggers is to check whether the users' weights (vertex weights) and weights of their connections (edges and hyperedges) are consistent. In our case, whether they express a consistent model with weighted securities of users and their connections. On *a scheduled trigger* the structure is inspected. Group leader issues an *optimization message*, which propagates through the structure and users optimize their adjecent sets in the terms of space utilization and computation load. An *on event trigger* is run whenever a new edge has been added into the structure. The action is reported to the corresponding GL by *an information message*.

An *on user request trigger* is a way how a user can influence the hypergraph structure more globally. A user can predefine some actions which should be treated differently then defined for the whole group. Another reason to start this trigger may be the need to modify asset numbers of some of users which influence the user. In the other words it enables a personalization in the structure.

4.5. Sharing

This subsection explains some issues concerning sharing and communication among users. Communication is done through message interchanging. The *target user id* is the unique group identifier of the user that is asked for data. Second value identifies the data demander. Demander asset number and edge's weight are also included. The *link type* cell holds information either it is transitive or direct, over-board or inter-group link respectively.

As given above, each user may store up to k level of adjacent neighbors. Clearly, if $k = 1$ then the communication complexity

$$2 * d(K), \tag{1}$$

where $d(K)$ is a diameter of the group K (which may consist of more components of the acyclic hypergraph). Space complexity is

$$|E| = \sum_i d_{out}{}^{(i)}. \tag{2}$$

If $k = 2$, communication load is

$$d(K) \tag{3}$$

and space complexity is

$$|E| = \sum_{(i,j)\in E(K)} d_{out}{}^{(i)}.d_{out}{}^{(j)}. \tag{4}$$

Each user has a *sharing value* set for sharing data. In our case, we use an integer value denoting a threshold under which sharing information with other users is rejected.

The following three rules controlling the sharing feature are defined in our implementation. The first rule is called *Direct*. It is useful in cases where data sharing is requested by users from an adjacent set. In such case direct connection weight and user's asset number are taken and compared to sharing value. The second rule will be called *friend-of-my-friend*. In such case there were no direct connections between users marked 1 and n. Therefore the message was released and users were forwarding it since the destination, user n, was found. Consequently the minimal edge's weight of all possible paths is taken and compared to sharing threshold. The third rule we call the *over-border* rule. This rule is based on the communication via appropriate group leaders. This procedure offers a possibility to join groups with common interests from different groups.

4.6. Security questions

Up to now we have discussed mostly mathematical and technical questions of a network of users. There are some global security questions which should be satisfied by any useful model. One of the biggest threat in security area is misapplication of private information (e.g. credit card number, personal identification number). Namely, if some secure information have been stolen from their proper holder, it can be consequently misused. Potential damages can be minimized by a time stamp validation. Since time stamps and invalidation times are part of GIN, both the proper user and the group leader/gate know time of the GIN invalidation. Further, the proper user and group leader/gate expect the time of the GIN invalidation and they also anticipate issue of a new GIN. Therefore only very rarely a user with stolen GIN, would try to access group with invalid GIN and can be easy revealed. The main question, which still remains, is how to treat with the GIN misuse while it is still valid? A possible solution lies in the underlying hypergraph structure. The structure is under continual evolution and it is optimized on every question issued by a group member (see section 4.3). Therefore the structure reflects favorite relationships between users. An example would be a good way to make it clear. Assume that a user stole a GIN and he/she is about to download as most as she/he could till the stolen GIN become invalid. In such case the "bad" user would ask as many group participant as he/she could and would demand data. The hypergraph structure - however, reveals such misuser very quickly. Once a group member behavior is found very different from its standard behavior, the user is disconnected from the network and a new GIN is consequently issued to the proper user. The time stamp validation time and the behavior check based on the underlying graph structure will significantly improve security issues.

5. Experimental Implementation

As the precedent was manly devoted to theoretical issues related to communication/computing in the large network of users, the implementation section makes the proposals alive. While the target environment might change a lot, as was mentioned earlier, an experimental application, called SECMOBILE, must be designed and created with respect to this. ANSI C language offers us a great opportunity to re-use existing implementations of both numerical and distributed computing paradigm with very optimized code as an imported DLL. Although the application is currently written as a console application without any graphic interface, in the future it will be given by a user friendly one.
The aim of the experimental application is to prove the correctness of the proposed algorithms and refine them if necessary. By the prove we will consider behavior of the network to not degenerate into one of the limiting cases; many groups consisting only one node; or few groups covering all nodes.
The implementation is based on a dynamic system of structures describing non zero values in the "adjacency" matrix. Adding new edges and vertices into the existing structure is maintained through binary or text files. Text files are designed in CSV fashion with a space separator. Since the structure should reflex usual behavior of users, creating such network cannot be held by random generation of nodes and/or vertices. An input file should be well formed taking into consideration "probable behavior" of a user.
While the proposed methodology keeps into consideration the complete spectrum of possibilities, the experimental application employs only a part. Its target is a structure creation and related operations. Higher levels of abstraction (e.g. sharing, user invitation process) will be considered and added consequently.

6. Conclusions

The paper deals with the security issues from a viewpoint of a distributed and self-evolving application. The design is mathematically sound being based on the weighted hypergraph model. The low level algorithms use supernode merging and splitting which was implemented in numerical linear algebra. The model is distributed and the local space and time complexities are very low. We discusses the relation among graph items and real-world application. The most important question whether the model can be consistently developed and model secure communication among users and their groups was answered positively in our implementation. Although we explained the model in the environment of mobile computing and communication, it can be easily generalized for some other application areas.

References

[1] S. Nanda, D.J. Goodman, "Dynamic Resource Acquisition in Distributed Carrier Allocation for TDMA Cellular Systems", *Proceedings GLOBECOM*, pp. 883–888, 1991.

[2] S. DasBit and S. Mitra, "Challenges of computing in mobile cellular environment a survey", *Elsevier B.V.*, 2003.

[3] A. Flaxman, A. Frieze, E. Upfal, "Efficient communication in an ad-hoc network", *Elsevier*, 2004.

[4] S. Basagni, "Remarks on Ad Hoc Networking", *Springer-Verlag*, Berlin Heidelberg, 2002.

[5] R. Molva, P. Michiardi, "Security in Ad Hoc Network", *IFIP International Federation for Information Processing*, 2003.

[6] Y. Lu, B. Bhargava, W. Wang, Y. Zhong and X, Wu, "Secure Wireless Network with Movable Base Stations", *IEICE Trans. Community*, vol. E86-B, 2003.

[7] Y. Zong, B. Bhargava and M. Mahoui, "Trustworthiness Based Authorization on WWW", *IEEE Workshop on Security in Distributed Data Warehousing*, 2001.

[8] J. Park, R. Sandhu and S. Ghanta, "RBAC on the Web by Secure Cookies", "Database Security XIII: Status and Prospects", Kluwer 2000.

[9] R. S. Sandhu, E. J. Coyne, H. L. Freinstein and C. E. Youman, "Role Based Access Control Models", *IEEE Computers*, Volume 29, 1996.

[10] P. K. Behera, P. K.Meher, "Prospects of Group-Based Communication in Mobile Ad hoc Networks", *Springer-Verlag Berlin Heidelberg*, 2002.

[11] P. O. de Mendez, P. Rosenstiehl, P. Auillans and B. Vatant, "A mathematical model for Topic Maps", *Springer-Verlag*, Berlin Heidelberg, 2002.

[12] M.C. Golumbic, "Algorithmic graph theory and perfect graphs", *Academic Press*, 1980.

[13] N. Selvakkumaran and G. Karypis, "Multi-objective hypergraph partitioning algorithms for cut and maximum subdomain degree minimization", *IEEE Transactions on Computer-aided design*, 2005, to appear.

[14] A. George and J.W.H. Liu, "A fast implementation of the minimum degree algorithm using quotient graphs", *ACM Trans. Math. Software*, 6(1980), 337–358.

[15] A. George, J.W.H. Liu, "Computer Solution of Large Sparse Positive Definite Systems", *Prentice Hall*, 1981.

Ústav Informatiky AV ČR
DOKTORANDSKÝ DEN ’05

Vydal
MATFYZPRESS
vydavatelství
Matematicko-fyzikální fakulty
University Karlovy
Sokolovská 83, 186 75 Praha 8
jako svou *– not yet –* . publikaci

Obálku navrhl František Hakl

Z předloh připravených v systému LaTeX
vytisklo Reprostředisko MFF UK
Sokolovská 83, 186 75 Praha 8

Vydání první

Praha 2005

ISBN *– not yet –*