Network-centric Access Control: Models and Techniques

Ting Wang* Mudhakar Srivatsa† Dakshi Agrawal‡

September 21, 2010

Abstract

In both commercial and defense sectors a compelling need is emerging for rapid, yet secure, dissemination of information to the concerned actors. Traditional approaches to information sharing (such as Multi-Level Security (MLS)) adopted a *node-centric* model wherein each user (social subjects) and each object (information object) is treated in isolation (e.g., using clearance levels for subjects and sensitivity levels for objects in MLS). Over the last two decades information sharing models have been enriched to partially account for relationships between subjects (e.g., Role-based Access Control (RBAC)), relationships between objects (e.g., Chinese-wall model), and relationships between subjects and objects (e.g., Separation of Duty (SoD) constraints).

In this paper, we present a novel *network-centric* access control paradigm that explicitly accounts for network-effects in information flows, and yet offers scalable and flexible risk estimation regarding access control decisions. The goal of this paper is not to prescribe a risk-model for information flows; instead we enable a class of risk-models by developing scalable algorithms to estimate *prior* and *posterior* information flow likelihood using the structure of social and information networks. For instance, our network-centric access control model answers questions of the form: Does subject s already have access (via her social network) to object o? If subject s is given access to object o, what is the likelihood that subject s' learns object o' (where the subjects s and s' are related via the social network and the objects o and o' are related via the information network)?

This paper makes three contributions. First, we show that several state-of-the-art access control models can be encoded using a network-centric access control paradigm, typically by encoding relationships as network edges (subject-subject, object-object and subject-object). Second, we present a suite of composable operators over social and information networks that enable scalable risk estimation for information flows. Third, we evaluate our solutions using the IBM SmallBlue dataset that was collected over a span of one year from an enterprise social network of size over 40,000.

1 Introduction

The emerging popularity of *social network systems* (SMS) has profound influence over today's information collecting, processing and disseminating infrastructures. As an example, the global social networking website FACEBOOK now has more than 300 million active users; 50% of active users log on the website in any given day; and among them, more than 2 billion pieces of contents (web links, news stories, blog posts, notes, photos, etc.) are shared each week [2]. Information sharing infrastructures constructed atop such social networking systems pose a set of critical challenges for access control mechanisms: 1) the access-control decision should now take account of underlying networking relationships among subjects and objects in estimating access risks; 2) the access risk estimation should be sufficiently computationally efficient in order to handle massive volume of access requests in a stream manner; 3) the estimation should also take into consideration the evolution of underlying social or information networks.

*Georgia Institute of Technology {twang@cc.gatech.edu}
†IBM T.J. Watson Research Center {msrivats@us.ibm.com}
‡IBM T.J. Watson Research Center {agrawal@us.ibm.com}

In facing such challenges, the traditional *node-centric* access control paradigm, however, has demonstrated its severe limitations. For example, the grant of access privilege should no longer be solely determined by simple comparison between the classification of the object o and the trustiness of the subject s; rather, one has to take account of the networking relationships among s and relevant subjects $\mathcal{N}(s)$, and that among o and relevant objects $\mathcal{N}(o)$:

- 1) prior information flow estimation: does s already have potential access to o through the channels between s and $\mathcal{N}(s)$ and that between o and $\mathcal{N}(o)$?
- 2) posterior information flow estimation: how would the decision of granting or denying access of o to s potentially affect the access privileges with respect to objects $\mathcal{N}(o)$ and subjects $\mathcal{N}(s)$?
- 3) evolved information flow estimation: how would the evolution of social or information network possibly affect the access risks?

The node-centric access control paradigm are inherently unable to answer these questions.

Therefore, in this paper, we advocate a shift from the *node-centric* paradigm to a *network-centric* paradigm: 1) we use networks to model both the relationship among objects (information network) and that among subjects (social network), and the inter-network links between information and social networks indicate the access-control level between objects and subjects; 2) on deciding granting or denying an request of accessing o from s, the networking influence from (and to) $\mathcal{N}(s)$ and $\mathcal{N}(o)$ before (and after) granting or denying this request is carefully evaluated; 3) the evolution of information or social network (e.g., the classification level of an object o degrades, or the relationship between two subjects s and s' changes) propagates through both intra-network and inter-network links. Equipped with such enriched semantics, this new paradigm is able to perform prior, posterior and evolved information flow estimation.

While enabling the network-centric paradigm to desirably capture the networking influence of both information and social networks, the enriched semantics also brings non-trivial technical challenges: for real-life networks comprising of thousands or even more nodes, 1) how to efficiently evaluate the networking influence of $\mathcal{N}(s)$ and $\mathcal{N}(o)$ to s and o? 2) how to efficiently update the access level with respect to $\mathcal{N}(s)$ and $\mathcal{N}(o)$ after a request to o from s is granted? 3) how to efficiently propagate the evolution of a part of the network to the whole network?

We address this set of scalability problems in a fully distributed manner. We present a suite of composable operators over social and information networks that enable scalable risk estimation for information flows. We show that, both theoretically and empirically, our solutions are distributed, scalable, and quality-guaranteed. It is worth emphasizing that all our experiments were performed over two real datasets, one corresponding to the social network of a subset of IBM employees who participated in the SmallBlue project [18], which involving more than 40,000 individuals, the other corresponding to the information network of archived bookmarks tagged by the individuals appearing in the first dataset, which contains 20,870 bookmark records, relevant to 7,819 urls.

The remainder of the paper is organized as follows. Section 3 formalizes the model of network-centric access control and the library of basic operations. Section 4 details our solution to implementing the basic operators. Section 5 introduces a set of operations that can be readily constructed by composing the set of basic operators. An experimental evaluation regarding our solution is presented in Section 7. The paper is concluded in Section 8.

2 Background

In this section, we give an overview of our network-centric access control model. We start with introducing fundamental concepts underlying risk-based access control, present the variety of traditional node-centric models, and discuss their limitations in facing the ever increasing complexity of social and information networks. We then present the mathematical model of network-centric access control model and the set of basic operations. Finally, we show how the variety of traditional models can be expressed in this new framework.

2.1 Node-Centric Access Control

Access control is a mechanism used to manage the leakage of sensitive information through human users in information systems. One of its major design objectives is to balance the need of the information consumers in order to perform their jobs and the need of the information owner to protect their sensitive information. An access control system concerns about administrating individuals' access behavior to its managed information. We term the individuals who request access as *subjects*, and the requested information as *objects*. Without loss of generality, we model subjects and objects using discrete sets $\mathcal{S}$ and $\mathcal{O}$; we use $\mathtt{q}(s \to o)$ to denote the request of a subject $s \in \mathcal{S}$ to access an object $o \in \mathcal{O}$. An access control policy encodes the rules used to evaluate the qualification of s regarding o. In composing access control policies, conventional models view s and o as individual nodes, i.e., *node-centric* access control.

In its simplest form, node-centric access control can be enforced as multilevel security model (MLS), e.g., Bell-La Padula model [6]. Each object o is associated with a classification level $\mathtt{class}(o)$ indicating its confidentiality; while each subject s is associated with a clearance level $\mathtt{clear}(s)$ indicating her trustworthiness level. The binary "allow/deny" decision can be statically and independently determined for each object o and subject s by comparing their corresponding confidentiality and clearance level.

The increasing complexity of users' information needs necessitated understanding the roles of different users and the relationships among these roles. An alternative model, role-based access control (RBAC), has been proposed. In its typical form, RBAC encodes the relationships among roles using *role hierarchies*, and places restrictive rule on the potential inheritance of permission from opposing roles using *constraints*. Here, the clearance level $\mathtt{clear}(\cdot)$ is essentially the partial order encoded by the role hierarchy and constraint. The use of RBAC to manage user privileges within a single system has been widely accepted, e.g., Microsoft SQL Server, FreeBSD and Solaris.

Meanwhile, the increasing complexity of information objects spurred the research on understanding and modeling relationships of objects in access control. A variety of well known models, e.g., Chinese Wall [9], were introduced. These models were designed to provide control that mitigate conflict of interest in commercial organizations; that is, no information can flow between the subjects and objects in a path that would create a conflict of interest. In such models, the classification level $\mathtt{class}(\cdot)$ is dynamically determined.

More recently, fuzzy logic-based MLS [10] and risk-based information sharing [21] frameworks have been proposed to support risk-based access control, which are designed to adapt the classification level $\mathtt{c}\mathit{class}(\cdot)$ and clearance level $\mathtt{c}\mathit{clear}(\cdot)$ to the dynamic environment. In a typical setting, a subject is charged certain risk tokens for an access, based on the classification level of the object and the trustiness level of the subject; each subject is periodically (say, monthly) allotted a risk budged of certain amount of tokens. The onus is on the subject to best leverage the risk budget; fixed risk-budget also offers bounds on information leakage. It has also been pointed out that the risk estimation should take account of the past access (or leakage) behaviors of the subjects; towards this end, history-based access control (HBAC) model, e.g., [14], has been proposed that accommodates such history information.

Despite the plethora of node-centric access control models, none of the notions above simultaneously model the relationships among objects (information network), that among subjects (social network), and that between objects and subjects (access/leakage history). In this paper, we attempt to propose a general access control model that incorporates all these factors in a single framework, and provide better understanding regarding the trade-off between information need and risk control.

3 Network-Centric Access Control Model

This section presents an overview of our network-centric access control model. We start with introducing fundamental operations underlying this model, then show how the variety of conventional access control models can be expressed in the new framework, and finally sketch the implementation of network-centric access control.

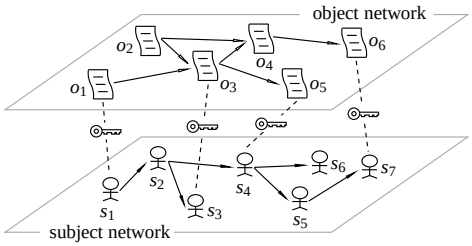

Figure 1: Model of information and social networks.

3.1 Fundamental Operations

The *network-centric* access control model revolutionizes conventional *node-centric* model by centering around the perspective of networking relationships among subjects and objects. The access risk estimation is no longer simply based on individual subject or object; rather, it depends on all (i) networked subjects, (ii) networked objects and (iii) relationships between them (e.g., social relationship, information inference relationship, and access relationship from subject to object). Figure 1 illustrates the model of information and social networks. Given such multi-layered network as context, evaluating the risk of an access request of a specific subject regarding a specific object should take into consideration the profound networking influence from (and to) relevant subjects and objects before (and after) granting the access privilege. Towards this end, we propose the following two fundamental operations, *prior*- and *posterior*-flow estimation.

Prior-Flow Estimation Prior to deciding on an access request $(s \to o)$, we intend to evaluate the latent information leakage risk[1] from o to s before the access is granted, via their networked objects and subjects. In the following, we use $f(o \to s)$ to denote the latent information flow (leakage) from object o to subject s. Prior-flow estimation provides information regarding the risk of such latent leakage; if the risk is already above certain threshold, it might be more suitable to grant the access than to deny it for saving on access control resources. Conceivably, in addition to directly evaluating the qualification of the subject respect to the object, prior-flow estimation provides an important risk-based exception handling mechanism.

For example, in Figure 1, on evaluating $(s_5 \to o_1)$, one may notice that a 2-hop neighbor (friend-of-friend) of s_5, s_6, has access (indicated by the inter-network link $\overline{o_5 s_6}$) to an object o_4, which is semantically proximate to o_1 (indicated by the inference relationship $\overline{o_1 o_4}$). The path $o_1 \to o_4 \to s_6 \to s_5$ may carry sufficient information for s_5 to completely infer o_1, which makes direct comparison of $\texttt{class}(o_1)$ and $\texttt{clear}(s_5)$ non-informative.

Posterior-Flow Estimation While prior-flow captures the potential leakage before an access $(s \to o)$ is granted, posterior-flow estimation evaluates once granted, how this access would impact the information flow $f(o' \to s')$ for objects $o' \in \mathcal{N}(o)$ relevant to o, and subject $s' \in \mathcal{N}(s)$ relevant to s. Formally, let $\underline{f}(o' \to s')$ and $\overline{f}(o' \to s')$ denote the prior and posterior information flows, before and after access $(s \to o)$ is granted, respectively. A large difference between $\overline{f}(o' \to s')$ and $\underline{f}(o' \to s')$ implies that the access would significantly affect the flow $f(o' \to s')$. Hence, if s' is restricted to access o', the grant of $(s \to o)$ results in a violation. Essentially, posterior-flow estimation measures the potential risk of approving an access request.

As an example, in Figure 1, the approval of access $(s_3 \to o_4)$ may significantly change flow $f(o_6 \to s_6)$, given the close social relationship between s_6 and s_3, and the inference relationship between o_6 and o_4. If $\texttt{clear}(s_6) < \texttt{class}(o_6)$, the increased leakage results in a severe violation of access control policy.

Also note that social and information networks are subject to frequent updates, i.e., network evolution [16], which may easily invalidate posterior-flow estimation based on existing social and information networks. It is, therefore, necessary to equip posterior-flow estimation with the capability of incorporating predicted evolution in risk-estimation process, which we refer to as *evolved-flow estimation*.

[1]In the following, without ambiguity, we use "information flow" and "information leakage" exchangeably.

3.2 Modeling of Networks

In the following, we introduce the mathematical modeling of basic elements of network-centric access control. We use networks to model both the relationships among objects (information network) and that among subjects (social network), and the inter-network links between social and information networks to model the relationships between subjects and objects (e.g., access or leakage history).

Specifically, the information network is represented as a directed graph $\mathcal{G}_I = (\mathcal{V}_I, \mathcal{E}_I)$ with the vertices $\mathcal{V}_I$ and edges $\mathcal{E}_I$ representing the set of objects and their relationships, respectively. Each edge is associated with type information indicating the relationship between the adjacent objects. Concrete examples include "mutually exclusive" relationships as in Chinese-wall policy, and semantic similarity (distance) indications between two objects.

Similarly, the social network is modeled as a direct graph $\mathcal{G}_\mathcal{S} = (\mathcal{V}_\mathcal{S}, \mathcal{E}_\mathcal{S})$, with $\mathcal{V}_\mathcal{S}$ and $\mathcal{E}_\mathcal{S}$ representing the set of subjects and their social connections, respectively. We assume that each relationship is associated with a type. More specifically, it could be assigned from a finite set of relationship types, e.g., {friend-friend, advisor-advisee, employer-employee}, or indicates the interaction of subjects, e.g., they belong to the same corporate. Each relationship is also associated with a *disclosure rate* indicating the possibility that one subject shares (leaks) its information with the other. Conceivably, this leakage rate is correlated with the corresponding relationship type; closer relationship tends to imply higher chance of leakage. For example, in a cover story of Reuters, the wife of the head of Britain's spy agency posted information of her husband, family and friends on FACEBOOK, details which could compromise security [1]. In general, such disclosure rate might be discriminative with respect to specific objects and subjects under consideration, i.e., a function of information metadata [22] and the corresponding social relationship type. The exact modeling of leakage rate is beyond the scope of this paper.

Each inter-network link between a subject s in the social network and an object o in the information network encodes (i) the clearance/classification level of s regarding o, and (ii) the access/leakage history of s regarding o. Note that in our model the clearance/classification level is dynamically determined based on the characteristics of o and s, and the networking influences from objects $\mathcal{N}(o)$ relevant to o and subjects $\mathcal{N}(s)$ relevant to s.

3.3 New Looks of Conventional Models

Next, we show how a variety of representative conventional access control models can be expressed under the general framework of network-centric access control. Note that we are not arguing to completely replace conventional node-centric models with network-centric model; rather, we believe that conventional access control models would be greatly enhanced under this general framework in terms of access risk estimation. Due to space constraints, we use three well-known node-centric models as concrete examples to show the compatibility of our network-centric paradigm with conventional node-centric models; more complicated models, e.g., FuzzyMLS [10] can also be readily composed under this general framework, with details referred to our technical report [5].

Multi-Level Security (MLS) Model Here, we use Bell-LaPadula (BLP) model, a classic MLS access control model, as a concrete example. In its simplest form, the policies in BLP are described by two terms, the *security attributes* of the objects concerned and the rules for access, in respect of *simple-security* and **-property*. BLP attaches security labels to both objects (classification level) and subjects (clearance level); the clearance/classification scheme is described in terms of a lattice. BLP also has a simple-security and a *-property rule, which can be characterized as "no read up, no write down":

- Simple security. Read access is granted only if the subject's clearance is above the object's classification.
- $*$-property. Write access is granted only if the subject's clearance is below the object's classification.

Under the network-centric framework, the implementation of BLP is straightforward:

- 1) The social and information networks encode the hierarchies of clearance/classification levels of all subjects and objects, respectively.
- 2) The prior-flow estimation always returns 0.
- 3) The posterior-flow estimation function returns 0 if $\texttt{class}(o) \leq \texttt{clear}(s)$ or 1 otherwise.
- 4) The risk estimation for an access request $(s \to o)$ raises an alarm on $\overline{f}(o \to s) - \underline{f}(o \to s) = 1$ for a read access, on $\overline{f}(o \to s) - \underline{f}(o \to s) = 0$ for a write access.

Role-based Access Control (RBAC) Model A standard RBAC [13] model uses the following conventional notations: $\mathcal{S}$, the set of subjects, $\mathcal{R}$, the set of *roles*, which describe authority levels, and $\mathcal{P}$, the set of permissions, which represent the approval of access to specific objects. Access control policy can be described by the following three mappings:

- Subject assignment $SA \subseteq \mathcal{S} \times \mathcal{R}$ which is a many to many subject to role assignment relation;
- Permission assignment $PA \subseteq \mathcal{P} \times \mathcal{R}$ which is a many to many permission to role assignment relation;
- Role hierarchy $PH \subseteq \mathcal{R} \times \mathcal{R}$ which is a partially ordered role hierarchy. Two elements $x \geq y$ means x inherits the permissions of y.

Under the network-centric model, we intend to encode these three mappings in the social and information networks and the inter-network relationships. One implementation could be as follows:

- 1) In the social network, in addition to the set of subjects $\mathcal{S}$, for each role $r \in \mathcal{R}$, we create a corresponding node n_r. For each $s \in \mathcal{S}$, a link $\overrightarrow{n_r s}$ indicates that s is assigned the role r, i.e., SA mapping. The sub-network over the node set $\{n_r\}_{r \in \mathcal{R}}$ encodes the partially ordered role hierarchy, i.e., PH mapping: two nodes n_r and $n_{r'}$ are adjacent over the link $\overrightarrow{n_r n_{r'}}$ iff r and r' are adjacent over the link $\overrightarrow{rr'}$ in PH.
- 2) The information network is a node set $\mathcal{O}$, each $o \in \mathcal{O}$ corresponding to an object.
- 3) The inter-network relationships between the information network and the sub-network over $\{n_r\}_{r \in \mathcal{R}}$ encode the permission assignment PA. Each link $\overrightarrow{on_r}$ between o and n_r indicates that the role r has access to o, and the specific access mode is contained in the type information of $\overrightarrow{on_r}$.
- 4) The flow (both prior and posterior) estimation function $f(o \to s)$ returns 1 if there is a directed path (the type of inter-network link must be equivalent to the requested access mode) from o to s in this two-layered network. Note that the posterior-flow estimation always returns 1 under this setting.
- 5) The risk estimation for an access request $(s \to o)$ raises an alarm on $\overline{f}(o \to s) - \underline{f}(o \to s) = 1$.

Chinese-wall and Variants Unlike other conventional access control models, Chinese-wall model [9] and its variants futher take into consideration temporal information, such as the access history of subjects regarding the objects concerned. In a simplified Chinese-wall model, each object o is associated with two label x_o indicating the commercial database holding o, and y_o indicating its *conflict of interest class*. The basic Chinese-wall policy essentially states that subjects are only allowed to access object that is not held to any other objects they have accessed. The access control policies can be described as:

- Simple security. An access $(s \to o)$ is granted only if o has the same label x_o as an object o' already accessed by s, i.e., within the wall, or has an entirely different label y_o to all the objects already accessed by s.

- $*$-property. Write access is granted only if the simple security rule is honored, and no accessible object o' contains unsanitized information and has a different label $x_{o'}$ to the requested one o.

Under the network-centric framework, one implementation of Chinese-wall model could be as follows:

- 1) The social network is a set of nodes $\mathcal{S}$, each corresponding to a subject.
- 2) In the information network, a pair of objects o and o' are adjacent iff (i) $x_o \neq x_{o'}$ and (ii) $y_o = y_{o'}$.
- 3) Once a subject s has accessed an object o, an inter-network link $\overline{so}$ is added to the two-layered network.
- 4) The flow (both prior and posterior) estimation function $f(o \to s)$ returns 1 if there is a path between o and s, and 0 otherwise. Note that the posterior-flow estimation always returns 1 under this setting.
- 5) The risk estimation for an access request $(s \to o)$ raises an alarm on $\overline{f}(o \to s) - \underline{f}(o \to s) = 0$. Further, if it is a write access, an alarm is raised if there exists an object o' such that $f(o' \overline{\to} s) = 0$ and $x_o \neq x_{o'}$.

3.4 A Library Like Implementation

In this paper, we provide a comprehensive library of operations to support the network-centric access control paradigm. In a nutshell, this library of operations essentially support estimating latent information flow in existing networks, updating intra-/inter-network links, and adding new network nodes. It is worth emphasizing that this library is not meant to be complete; nevertheless, we intend to utilize this library to show the rich expressiveness power of the network-centric access-control paradigm. As we will reveal in the next section, these set of operations are implemented by composing a set of even more fundamental operations, called *atom operators*, which are neutral to the types of networks. We will show that our library is readily extendible and new operations can be constructed atop the atom operations. In the next section, we first introduce the set of atom operators that serve as the building blocks of our library.

4 Atom Operators

In this section, we detail our implementation of the basic operation defined in Section 3. The objectives Our design carries the following key objectives. 1) Scalability. The operations should be able to support large-scale networks (up to thousands of nodes), and handle stream-manner updates. 2) Modularity. The implementation should be self-inclusive and reusable by different operations, and even supports operations to be defined. We achieve this by defining a set of *atom networking operations* which are composable to form higher-level operation. Note that this set of atom operations are neutral to the types of underlying networks; they are therefore interesting in their own right.

4.1 Operations for Static-Flow Network

We start with discussing the atom operations for Information Network, which is assumed to be a *static-flow* network. By "static-flow", we mean that once the network is fixed, including the topology of the network and the properties of vertices and edges, the information flow in the network is static. In Section 4.2, we will contrast this concept with a dynamic-flow network, e.g., social network, where the information flow tends to change depending on nodes' dynamic behaviors. The distinct characteristics of static-flow and dynamic-flow networks necessitate different treatment in measuring information flows.

In an information network $\mathcal{G}_I = (\mathcal{V}_I, \mathcal{E}_I)$, for a given object i, we are particularly interested in studying the *residual information* of i in another object j, or a set of objects $\mathcal{J}$, formally

Definition 1 (Residual Information)**.** *With respect to a given object i, the residual information of i in another object i, r_{ij}, is defined as the probability of i that is inferable from j.*

Clearly, one can have a variety of instantiations of this abstract definition. For example, one can define r_{ij} base on the Kullback-Leibler divergence between i and j. We do not make specific assumption regarding the instantiation, but assume that r_{ij} is a real number within the interval [0, 1].

The concept of residual information can be generalized to a set of objects with respect to a source i; in particular, we define the operation of union, denoted by $\oplus$, for a set of objects. The residual information of i at a set of objects $\mathcal{J}$, $r_{i\mathcal{J}}$, is formalized as follows:

$$r_{i\mathcal{J}} = \bigoplus_{j \in \mathcal{J}} r_{ij}$$

Here, the $\oplus$ operation combines all the information appearing in the objects in $\mathcal{J}$. In real applications, it might not be feasible to directly calculate this quantity; instead, we are interested in establishing its upper and lower bounds. A straightforward bound for $r_{i\mathcal{J}}$ might be

$$\max_{j \in \mathcal{J}} r_{ij} \leq r_{i\mathcal{J}} \leq \min\{\sum_{j \in \mathcal{J}} r_{ij}, 1\} \tag{1}$$

Intuitively, the residual information about i in a set of objects $\mathcal{J}$ is above the maximum residual information in any single object in $\mathcal{J}$, while is also below the sum of the residual information in all the objects of $\mathcal{J}$. Later, we will refine this bound by incorporating network structure information.

Next, we proceed to discussing the calculation of r_{ij} for different network topologies. First, we need to re-formalize the concept of information network from the perspective of a specific object i.

Definition 2 (Viewpoint Information Network)**.** *A viewpoint Information Network $\mathcal{G}_I^i = (\mathcal{V}_I^i, \mathcal{E}_I^i)$ with respect to a node i is a sub-graph of the information network $\mathcal{G}_I$. It is constructed as follows: each vertex $j \in \mathcal{V}_I^i$ represents an object with non-zero residual information of i, i.e., $r_{ij} > 0$; a directional edge $\overrightarrow{jk}$ represents the dependency of k on j with respect to the information regarding i, and the weight w_{kj} indicates the fraction of residual information of i at j passed through the dependency to k*[2]*.*

While the information network $\mathcal{G}_I$ could be of arbitrary topology, we assume that a viewpoint information network $\mathcal{G}_I^i$ is a *directed acyclic graph*, where the directed edges encode dependency relationships among objects with respect to residual information of i. Such dependencies prevail in real networks, in the form of *temporal* dependency, *functional* dependency, etc. For a concrete example, in a blog network, each node could represent a blog, while a link could represent the reference between two blogs. Further, we can consider the edge weight as an indication of *information degradation*, i.e., the information originated at a source object gradually fades as it passes along the edges in the network.

In its simplest form, $\mathcal{G}_I^i$ could be a chain. In this case, the estimated residual information of i at an object j is trivial, i.e., the information originated at an object i gradually fades as it passes along the edges in the network. Let $(v_0, v_1, v_2, \ldots, v_n)$ denote the sequence of nodes between i and j (v_0 and v_n correspond to i and j, respectively), the residual information r_{ij} can be estimated as

$$r_{ij} = \prod_{i=0}^{n-1} w_{v_i v_{i+1}}$$

Now, let us consider the case that multiple paths exists between i and j. Figure 2 illustrates an example. The source object i passes its information through several overlapping paths to the destination object j. Before discussing in detail our estimation method, we begin with introducing a set of fundamental concepts.

Definition 3 (Ancestor)**.** *Given a viewpoint information network $\mathcal{G}_I^i$, an object k is an ancestor of another object j if there exists a directed path from k to j in $\mathcal{G}_I^i$. We use $\mathcal{A}_j$ to denote the set of all the ancestors of j; in particular, we use $\mathcal{P}_j$ to denote the set of direct ancestors (parents) of j.*

We further introduce the concept of *flow*.

[2]Without ambiguity, we omit the referred object i in the notation of w_{jk}.

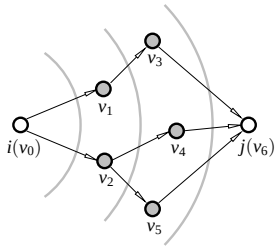

Figure 2: Information degradation in viewpoint information network.

Definition 4 (FLOW)**.** *In a viewpoint network $\mathcal{G}_I^i$, the flow of an directed edge $\overrightarrow{kj}$, f_{kj} is the residual information regarding i passes through $\overrightarrow{kj}$. Further, we use $\mathcal{PF}_k$ and $\mathcal{CF}_k$ to denote the parent (incoming) and child (outgoing) flows of the object k, respectively.*

For a given edge $\overrightarrow{kj}$, the residual information of i carried by the flow f_{kj}, f_{kj}, is estimated by:

$$f_{kj} = r_{ik} \cdot w_{kj}$$

$\mathbb{J}$-operator

To estimate the residual information at a given object, we introduce the *join* operator (J-operator) over a set of flows.

Definition 5 (JOIN)**.** *In a viewpoint network $\mathcal{G}_I^i$, for a given set of flows $\mathcal{F} = \{f_{k_1j_1}, f_{k_2j_2}, \ldots, f_{k_nj_n}\}$. The join operator $\oplus$ over $\mathcal{F}$, denoted by $\oplus\mathcal{F} = f_{k_1j_1} \oplus f_{k_2j_2} \oplus \ldots \oplus f_{k_nj_n}$, estimates the residual information of i in the union of these flow.*

Now, we are ready to formalize the problem of estimating the residual information r_{ij}; we re-define it in an iterative manner:

$$\begin{cases} r_{ij} = \bigoplus_{k \in \mathcal{P}_j} f_{kj} \\ f_{kj} = r_{ik} \cdot w_{kj} \end{cases} \tag{2}$$

Clearly, the J-operator is the key to estimating r_{ij}. Following, we detail its implementation. As we have discussed above, applying the J-operator over a set of flows $\mathcal{F}$ returns the estimation of the residual information of i in the union of these flows. First notice that we can establish the following bounds. Note that we use f to denote a flow and f its carried residual information.

$$\max_{f \in \mathcal{F}} f \leq \oplus\mathcal{F} \leq \min\{\sum_{f \in \mathcal{F}} f, 1\} \tag{3}$$

We note that the lower bound is tight, as proved in the next theorem:

Theorem 1. *The lower bound of $\oplus\mathcal{F}$, $\max_{f \in \mathcal{F}} f$, is tight.*

Proof. Without loss of generality, we use an interval to represent the residual information of the object i. Initially, the complete information is $[0, 1]$.

We construct a viewpoint information network $\mathcal{G}_I^i$ as follows. At each object k, we let the residual information of i passed to a direct descent j of k along the edge $\overrightarrow{kj}$ be $[0, r_{ik} \cdot w_{kj}]$.

It can be derived that under this setting, for any two flows, we can strictly compare them based on their residual information, i.e., one contains the other. Therefore, the union of these flows carries the same amount of information as the maximum flow $f^* = \arg\max_{f \in \mathcal{F}} f$. □

While the lower bound is tight, one can essentially obtain better upper bound based on the following observation: for two flows f and f' going out of an object k, clearly, the residual information of the union

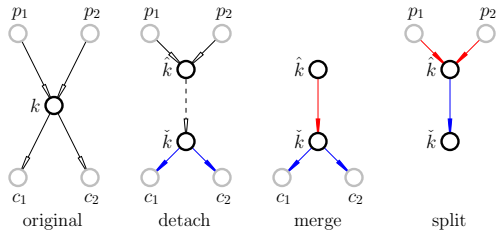

Figure 3: Operations for estimating effective flows, where the effective flows on blue links are materialized, and those on red links are being updated.

of f_{kj_1} and f_{kj_2} can not exceed that of k; hence, $f_{kj_1} \oplus f_{kj_2} \leq \min\{r_{ik}, f_{kj_1} + f_{kj_2}\}$. We intend to establish tighter upper bound by exploiting this observation.

We would like to contrast the J-operator with that of estimating maximum flow in a regular flow network. (1) In a regular flow network, each link is associated with a *capacity*, the maximum allowed flow through this link; while in an information network, instead, each link is associated with a degradation rate. (2) In a flow network, one aims at maximizing the flow from a source to a destination; while in an information network, we attempt to estimate the *maximum residual information* of a source object at a destination object, based on the network topology. (3) The flows in our setting are *duplicable*. For an incoming flow to an object, one may create an identical copy of it for each outgoing link. (4) The flows in our setting are *joinable*; each incoming flow can be considered a subset of a universe, and a set of flows can be joined into a new flow. To the best of our knowledge, no previous work has studied the present problem.

We first introduce a fundamental concept, *cut*.

Definition 6 (Cut)**.** *In a viewpoint information network $\mathcal{G}_I^i$, for a set of flows $\mathcal{F}$ containing no inheritance relationship between any pair of flows, let all the flows in $\mathcal{F}$ inject into a virtual sink j. A set of edges are called a cover of $\mathcal{F}$ if they separate i and j.*

Intuitively, all the flows of $\mathcal{F}$ injected into j must go through a cut of $\mathcal{F}$; particularly, $\mathcal{F}$ itself is a cut for $\mathcal{F}$. Under this framework, estimating the upper bound of $\oplus\mathcal{F}$ can be re-formulated as finding a cut that carries the minimum *effective* flows with respect to $\mathcal{F}$. By effective flow, we refer to the amount of information that is responsible for generating $\mathcal{F}$. We use $\underline{f}$ to denote the effective part of a flow f. Clearly, for each $f \in \mathcal{F}$, $\underline{f} = f$. Next, we show how to estimate the upper bound of effective flow carried by each other edge.

We achieve this objective in a bottom-up manner; starting from $\mathcal{F}$, we chase back to the root i. At each step, we consider the following three operations, as illustrated in Figure 3:

- *detach*. For each object k encountered in the computation, we detach k into two nodes $\hat{k}$ and $\check{k}$, and connect them with an edge $\overrightarrow{\hat{k}\check{k}}$; $\hat{k}$ is connected to the parent flows of k while $\check{k}$ is connected to the child flows of k. The flow on $\overrightarrow{\hat{k}\check{k}}$, $f_{\hat{k}\check{k}}$, is set as r_{ik}.

- *merge*. Given a set of flows going out of an object k (now $\check{k}$), $\{f_{kc_1}, f_{kc_2}, \ldots, f_{kc_n}\}$ whose effective parts have been estimated, we update the effective flow of $\overrightarrow{\hat{k}\check{k}}$ using the following rule:

$$\underline{f}_{\hat{k}\check{k}} = \min\{\underline{f}_{\hat{k}\check{k}}, \sum_{l=1}^{n} \underline{f}_{kc_l}\}$$

  That is, the effective flow on $\overrightarrow{\hat{k}\check{k}}$ can not exceed the sum of the effective parts in all its child flows.

- *split*. Given a set of flows injected into an object k (now $\hat{k}$), $\{f_{p_1k}, f_{p_2k}, \ldots, f_{p_mk}\}$, we update the estimation regarding their maximum effective flows using the rule as follows:

$$\underline{f}_{p_lk} = \min\{f_{p_lk}, \underline{f}_{\hat{k}\check{k}}\} \quad (1 \leq l \leq m)$$

  Intuitively, the effective flow on each incoming link can not exceed that on $\overrightarrow{\hat{k}\check{k}}$.

Clearly, if the effective flows on all the links have been estimated, finding the upper bound of $\oplus\mathcal{F}$ is equivalent to finding a minimum cut of this *effective-flow* graph, where the weight of an edge is defined as its effective flow. Now, we show how to construct such an effective-flow graph using the operations we have defined above.

Algorithm 1 sketches the construction of J-operator. It mainly consists of two parts: (1) constructing the effective-flow graph, and (2) finding the minimum cut of the graph. It iterates over the source objects of the flows in $\mathcal{F}$ which is subjected to update. At each iteration, it picks the source object k with the largest topological order (the head of the list); it first detaches k, merges the flows going out of k in the current list $\mathcal{F}$, and then splits the flows injected into k; it then updates the set of source objects $\mathcal{S}$ and the current flow list $\mathcal{F}$. In the second phase, it invokes the process of finding the minimum $i-j$ cut of the network. It is noticed that in the first phase, the algorithm visits each flow (each link) at most once, and in the second phase, a max-flow min-cut procedure, e.g., Edmonds-Karp algorithm [12], features a complexity of $O(|\mathcal{V}_I^i| \cdot |\mathcal{E}_I^i|^2)$, which therefore dominates the overall complexity.

Input: viewpoint network $\mathcal{G}_I^i$, a set of flows $\mathcal{F}$
Output: upper bound of $\oplus\mathcal{F}$
create a virtual object j in $\mathcal{G}_I^i$ collecting all flows in $\mathcal{F}$;
$\mathcal{S} \leftarrow$ sources of $\mathcal{F}$;
`// topological sorting`
sort $\mathcal{S}$ in a decreasing order;
while $\mathcal{S} \neq \emptyset$ **do**
  $k \leftarrow$ pop the head of $\mathcal{S}$;
  `// detach + merge + split operation`
  detach k;
  $\mathcal{V}_I^i \leftarrow \mathcal{V}_I^i \setminus \{k\} \cup \{\hat{k}, \check{k}\}$;
  merge $\mathcal{CF}_k \cap \mathcal{F}$;
  split $\mathcal{PF}_k$;
  `// update` $\mathcal{S}$ `and` $\mathcal{F}$
  $\mathcal{F} \leftarrow \mathcal{F} \setminus \mathcal{CF}_k \cup \mathcal{PF}_k$;
  $\mathcal{S} \leftarrow \mathcal{S} \cup$ sources of $\mathcal{PF}_k$;
`// remove irrelevant objects`
remove from $\mathcal{G}_I^i$ all unvisited objects;
`// min-cut process`
find the minimum $i-j$ cut w;
output w as the upper bound of $\oplus\mathcal{F}$;

Algorithm 1: *J*-operator.

$\mathbb{E}$-operator

In addition to the upper and lower bounds of the union of a set of flows $\mathcal{F}$, we might also be interested in the expected value of $\oplus\mathcal{F}$. Unfortunately, exact evaluation of $\mathbb{E}(\oplus\mathcal{F})$ is equivalent to solving a maximization problem in a graphical model. It is known that performing exact or approximately (at least for relative error) inference in a graph beyond tree is NP-hard [7]. Here, we present a simple yet effective sampling based solution, which we refer to as the E-operator.

The construction of E-operator is sketched in Algorithm 2. First, for the set of flows $\mathcal{F}$ with no inheritance relationship, we create a virtual sink j, and attempt to estimate the residual information at j, $r_{ij} = \oplus\mathcal{F}$. One initializes a N-bit vector with all ones at i, and passes the vector through the network following a breath-first-search paradigm. At each node k, one unions the vectors sent through each incoming link as the residual information at k; one then duplicates the residual information for each outgoing link of k, flips certain one-bits according to the degradation probability, and passes the residual information to the corresponding descent. The amount of residual information at the target object j is estimated by the number of one bits received by j. It is clear that each edge is visited at most once following the breath-first-search paradigm, thus leading to the overall complexity as $O(|\mathcal{E}_I^i| \cdot N)$.

The following theorem establishes the bound of number of bits needed to guarantee sufficiently accurate estimation.

Theorem 2. *For a set of flows $\mathcal{F}$ with a virtual sink j, let r_{ij} and $\hat{r}_{ij}$ be the exact and estimated residual*

Input: viewpoint network $\mathcal{G}_I^i$, a set of flows $\mathcal{F}$, N
Output: expectation $\mathbb{E}(\oplus\mathcal{F})$
create a virtual object j absorbing $\mathcal{F}$;
initialize a N-bit one vector v_i at i;
$\mathcal{A} \leftarrow \{i\}$;
// breach-first-search
$k \leftarrow$ pop the head of $\mathcal{A}$;
while *true* **do**
  if k *has ancestor* **then** $\mathsf{v}_i \leftarrow N$-bit zero vector;
  for *each incoming link* $\overrightarrow{hk}$ **do**
    $\mathsf{v}_k = \mathsf{v}_k \cup \mathsf{v}_{hk}$;
  if $k = j$ **then** break;
  for *each outgoing link* $\overrightarrow{kh}$ **do**
    $\mathsf{v}_{kh} \leftarrow \mathsf{v}_k$;
    flip "one" bits in v_{kh} with probability $(1 - \mathsf{w}_{kh})$;
    pass v_{kh} to h;
    if h *has not been visited* **then** push h to $\mathcal{A}$;
  $k \leftarrow$ pop the head of $\mathcal{A}$;
output number of one-bits in v_j divided by $\mathcal{N}$ as $\mathbb{E}(\oplus\mathcal{F})$;

Algorithm 2: E-operator.

information, respectively. One needs $N \geq \frac{1}{2\epsilon^2}\ln(\frac{2}{\delta})$ *bits in order to guarantee that the probability that* $|\hat{r}_{ij} - r_{ij}| \geq \epsilon$ *lies below* δ.

Proof. Consider each bit x_i in the vector v_j as a random variable. Clearly, $\mathbb{E}(x_i) = r_{ij}$, and $x_i \in \{0, 1\}$. Therefore, applying the Hoeffding's inequality, we have

$$p[|\hat{r}_{ij} - r_{ij}| \geq \epsilon] \leq 2\exp(-\frac{2\epsilon^2 N^2}{N}) = 2\exp(-2\epsilon^2 N)$$

Let $p[|\hat{r}_{ij} - r_{ij}| \geq \epsilon] \leq \delta$, one can thus obtain the lower bound for N: $\frac{1}{2\epsilon^2}\ln(\frac{2}{\delta})$. □

4.2 Operations for Dynamic-Flow Network

Now, we proceed to examining the atom operators necessary to support the operation for dynamic-flow networks, e.g., social network. Compared with a static network, e.g., information network, a dynamic network demonstrates several critical features. 1) The information propagation behavior of each node is not deterministic; rather, it only statistically follows certain pre-defined formations. 2) The network structure is subjected to frequent updates, e.g., new individuals (subjects) are injected into the network, or the social relationships among existing individuals are frequently updated; clearly, the network evolution should be taken account in designing and implementing the atom operators.

We start with introducing the detailed modeling of social network. In contrast with the information network, the social network is a more "free-style" graph; for example, no strict dependency relationships exist in such a network. Formally,

Definition 7 (Social Network)**.** *We model a social network as a general graph* $\mathcal{G}_S = (\mathcal{V}_S, \mathcal{E}_S)$ *with* $\mathcal{V}_S$ *representing the collection of individuals (subjects) and* $\mathcal{E}_S$ *representing the social relationships among them. We assume that the relationship between any two individuals* i *and* j*, denoted by* r_{ij} *is drawn from a finite set* $\mathcal{R}$*. Also, note that the relationship between two individuals could be symmetric, e.g., friend-to-friend, or asymmetric, e.g., advisor-to-advisee, in which case,* $r_{ij} \neq r_{ji}$.

Further, we assume that each type of relationship $r \in \mathcal{R}$ is associated with a *information-disclosure possibility*, denoted by $\mathsf{weight}(r)$. For two individuals i and j with relationship r, $\mathsf{weight}(r)$ indicates the possibility that i passes its possessed information to j. Note that in this paper, we focus on studying the statistical behavior of the information propagation in the social network; hence, we define the leakage quantity as a statistical quantity.

Now, we intend to model the information propagation behavior at a specific individual i. It is noted that here we treat each piece of information as indifferent, while it is feasible to define finer granularity models

that take account of the classification levels of the information under consideration. Let $\mathcal{N}_i$ be the set of (outgoing) neighbors of the individual i. In addition, we consider a special neighbor, i itself.

For each $j \in \mathcal{N}_i$, let r_{ij} denotes the relationship type of the edge $\overrightarrow{ij}$. We specify that the probability that i passes the information through edge $\overrightarrow{ij}$ to another subject j, p_{ij} as:

$$p_{ij} = \begin{cases} \frac{weight(r_{ij})}{W} & (j \in \mathcal{N}_i) \\ \frac{W - \sum_{j' \in \mathcal{N}_i} weight(r_{ij'})}{W} & (j = i) \end{cases} \tag{4}$$

where W is a global constant, which is specified in this way such that $W \geq \max_{i \in \mathcal{V}_S} \sum_{j \in \mathcal{N}_i} weight(r_{ij})$. The quantity p_{ij} indicates the possibility that the information is propagated (leaked) from the edge $\overrightarrow{ij}$; p_{ii} presents the probability that i keeps the information to him/herself. Intuitively, if i has a number of "channels" to propagate the information, then the probability that i keeps the secrets to him/herself would be low. We will discuss in detail the optimal setting of W in Section **??**. It is noted that, under this model, for a specific type of relationship r, the probability of information propagation is fixed: $weight(r)/W$.

We attempt to estimate the probability that the information possessed by a subject i propagates (leaks) to another subject j also belonging to the social network $\mathcal{G}_S$. Note that it is infeasible to exactly measure this quantity, which depends on the subjects' personalized behaviors and the contents of the information; instead, we intend to study the statistical property of this quantity, based solely on the relationship types and the network topologies.

We apply the concept of *random walk* to capture the leaking probability; ideally, if two subjects have many common neighbors with good relationships, or they belong to a small yet tight community, the probability that the information propagates from one to the other would be high. A rather intuitive way of capturing this is the expected path length from one subject to the other during a random walk, which is called the *hitting time* [4]. It tells how long it takes on average to hit the target subject from the source subject. This measure is inherently robust to noise and exploits the inherent information encoded in the network structure. It has been successfully applied in a variety of applications, including ranking in recommender network [8], link prediction in social network [17], and image segmentation [19].

Nevertheless, for our purpose, hitting times suffer two major drawbacks, as observed in [17] : 1) they tend to be small whenever one of the nodes has a high outgoing degree, and 2) they are sensitive to parts of the graph that are far away from the nodes even when short paths between the nodes exist. Therefore, we are interested in a truncated variant of random walks [20], which emphasizes the influence of "short-range" neighbors. Specifically, a *T-truncated hitting time* considers only paths of length less than T.

Definition 8 (T-Truncated Hitting Time)**.** *The T-truncated hitting time between two nodes i and j, denoted by h_{ij}^T, measures the expected steps taken by a random walk starting from i to hit j for the first time. It can be defined in the following recursive form:*

$$h_{ij}^T = 1 + \sum_{k \in \mathcal{N}_i} p_{ik} \cdot h_{kj}^T \tag{5}$$

where h_{ij}^T is defined to be zero if $i = j$ or $T = 0$.

Similar to the static-flow network, here, we are interested in two types of atom operators, *merge* and *diffuse*. Intuitively, the merge operator calculates the propagation probability from all the possible sources to a given destination; while the diffuse operator measures the propagation probability from a given source to all the possible destinations.

$\mathbb{M}$-operator

In the context of social network, for a given target subject j, the M-operator measures the information propagation probability from a set of possible source subjects to j. More specifically, we define the propagation probability between two subjects i and j, $prop(i, j)$, reversely proportional to the hitting times between i and j, formally

$$prop(i, j) = \frac{T - h_{ij}^T}{T} \tag{6}$$

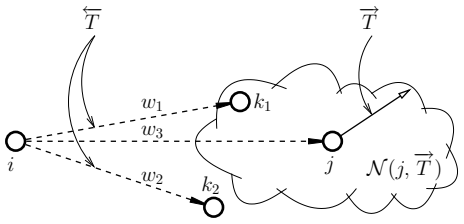

Figure 4: Decomposition of a walk from i to j.

Furthermore, we focus our attention to neighbors within L hops of j[3]. This formation therefore points to measuring the hitting times from the subjects within L hops to the target one j. Formally, let H^T denote the hitting-time matrix, where an entry (i, j) indicates the T-truncated hitting time between the subjects i and j. The M-operator for a target subject j essentially calculates the j-th column of H^T, denoted by $H^T_{\cdot j}$.

We construct the M-operator based on the formulation of Equation (5). Let P denote the transition matrix with each entry (i, j) as defined in Equation (4), and $\mathcal{N}^L_j$ be the neighbors within L hops of i. Algorithm 3 sketches the structures of M-operator: it iterates for T rounds, and computes $H^T_{\cdot j}$ based on $H^{T-1}_{\cdot j}$ and P.

Input: transition matrix P, target subject j
Output: j-th column of T-truncated matrix $H^T_{\cdot j}$
`// initialization`
if $i \neq j$ **then** $h^{old}_{ij} \leftarrow T$ **else** $h^{old}_{ij} \leftarrow 0$;
for h *from* 1 *to* T **do**
  for $i \in \mathcal{N}^L_j$ **do**
    `// iteration`
    $h^{new}_{ij} = 1 + \sum_{k \in \mathcal{N}_i} p_{ik} \cdot h^{old}_{kj}$;
  copy h^{new}_{ij} to h^{old}_{ij} for $i \in \mathcal{N}^L_j$;
set the i-th entry of $H^T_{\cdot j}$ as h^{new}_{ij} for $i \in \mathcal{N}^L_j$;

Algorithm 3: M-operator.

It can be proved that the computational complexity of Algorithm 3 is $O(T\Delta|\mathcal{N}^L_j|)$ where Δ is the average outgoing in the social network. We intend to cache partial reusable results for the D-operator that will be introduced next while also take account of frequent updates. To this end, after the calculation of $H^T_{\cdot j}$, we cache the results for subjects with hitting times below $\overline{T}$. This caching strategy enjoys a set of advantages: 1) the cached result is directly reusable to calculate H^T_j; it is insensitive to local change; further, we will show in Section **??** that this strategy is amenable to indexing and pattern-based acceleration.

The subjects with hitting times less than $\overline{T}$ to the target j are referred to as the *$\overline{T}$-incoming neighbors* of j, denoted by $\mathcal{N}(j, \overline{T})$, which will be reused in the construction of D-operator.

$\mathbb{D}$-operator

For a given source subject the D-operator essentially intends to evaluate the i-th row in the hitting-time matrix H^T, denoted as $H^T_{i\cdot}$. Similar to the M-operator, here, we pose the constraint that the paths only consist of nodes within L hops to the target subject.

In contrast of the M-operator, however, there is no straight solution to evaluate one row in the hitting-time matrix. In order to exactly evaluate $H^T_{i\cdot}$, one is potentially forced to compute the entire matrix H^T, which features the computational complexity af $O(|\mathcal{V}_S|^3)$.

Here, we introduce an efficient construction of D-operator based on a random sampling scheme and the cached result generated by the M-operator. Recall that for a given target subject j, the M-operator

[3]We pose this constraint to make both M-operator and D-operator computationally feasible. We need to give some explanations here.

generates and caches the truncated hitting times for the (incoming) neighbors with hitting times below $\overline{T}$ to j. Now, let us consider measuring the hitting time for a given source subject i to a given target j. Clearly, if i has the hitting time to j below $\overline{T}$, the information could be simply looked up in the cache; we focus on the case that $h_{ij}^{T} > \overline{T}$.

Let us consider a T-step walk w starting from i: we divide w into two parts, a part with length $\overrightarrow{T}$ which satisfies $\overrightarrow{T} + \overline{T} = T$ and the rest part with length $\overline{T}$. The following two cases apply: 1) w hits j in the first $\overrightarrow{T}$ steps; 2) w hits a node k ($k \neq j$) at the $\overrightarrow{T}$-step. We define the following hitting probability, p_{ik}^{t}: 1) if $k = j$, p_{ik}^{t} represents the probability that w hits j for the first time at the t-th step; 2) if $k \neq j$, p_{ik}^{t} indicates the probability that w does not hit j in the first $\overrightarrow{T}$ steps, and lands at k at the $\overrightarrow{T}$-th step. It is clear that the hitting probability satisfies that

$$\sum_{t=1}^{\overrightarrow{T}} p_{ij}^{t} + \sum_{k \neq j} p_{ik}^{\overrightarrow{T}} = 1$$

We can then reformulate Equation 5 in the following form:

$$h_{ij}^{T} = \sum_{t=1}^{\overrightarrow{T}} t \cdot p_{ij}^{t} + (1 - \sum_{t=1}^{\overrightarrow{T}} p_{ij}^{t}) \cdot \overline{T} + \sum_{k \neq j} p_{ik}^{\overrightarrow{T}} \cdot h_{kj}^{\overline{T}} \tag{7}$$

It is noted that in the formation above, the quantity $h_{jk}^{\overline{T}}$ has been generated an cached by the M-operator for all k: if $k \in \mathcal{N}(j, \overline{T})$, the result is cached; if $k \notin \mathcal{N}(j, \overline{T})$, it is regarded as $\overline{T}$. The key to this formation is therefore to estimate p_{ik}^{t} for every $k \in \mathcal{N}(j, \overline{T})$ and $t \in [1, \overrightarrow{T}]$. Next, we present a sampling scheme that achieve this with results reusable for all $j \in \mathcal{V}_S$. Algorithm 4 sketches the sampling process, which we refer to as the S-operator. Essentially, it runs N $\overrightarrow{T}$-step independent walks, and truncate a walk if it hits j; then, it counts the number of walks that hit a specific node, and the taken steps. Clearly, the set of walks $\mathcal{W}$ is reusable for all $j \in \mathcal{V}_S$.

Input: start node i, end node j, step limit $\overrightarrow{T}$, neighborhood $\mathcal{N}(j, \overline{T})$
Output: p_{ik}^{t} for every $k \in \mathcal{N}(j, \overline{T})$ and $t \in [1, \overrightarrow{T}]$
run N $\overrightarrow{T}$-step walks $\mathcal{W}$ starting form i;
for *each walk* $w \in \mathcal{W}$ **do**
  truncate w at hitting j;
// initialization
$h_{ij}^{t} \leftarrow 0$ for $t \in [1, \overrightarrow{T} - 1]$;
$h_{ik}^{\overrightarrow{T}} \leftarrow 0$ for $k \in \mathcal{N}(j, \overline{T})$;
// $e(w)$ denotes the last node of a walk w
for *each* $w \in \mathcal{W}$ **do**
  if $|w| = \overrightarrow{T}$ **then**
    if $e(w) \in \mathcal{N}(j, \overline{T})$ **then** $h_{ie(w)}^{\overrightarrow{T}} = h_{ie(w)}^{\overrightarrow{T}} + 1/|\mathcal{W}|$;
  else
    $h_{ik}^{|w|} = h_{ik}^{|w|} + 1/|\mathcal{W}|$;

Algorithm 4: S-operator.

Now, we provide the bound on the number of samples needed in order to obtain sufficiently accurate estimation.

Theorem 3. *For any node $k \in \mathcal{N}(j, \overline{T})$ and $t \in [1, \overrightarrow{T}]$, let $\hat{p}_{ik}^{t}$ be the estimation value of p_{ik}^{t}. One needs $N \geq \frac{1}{2\epsilon^2} \ln(\frac{2}{\delta})$ samples in order to guarantee that the probability of $|\hat{p}_{ik}^{t} - p_{ik}^{t}| \geq \epsilon$ lies below δ.*

Proof. Let X_r be a random variable such that $X_r = 1$ if the r-th walk hits k at the t-th step, or 0 otherwise. It is noted that $\mathbb{E}(X_r) = p_{ik}^{t}$ for $r \in [1, N]$ and $\hat{p}_{ik}^{t} = \sum_{r=1}^{N} X_r / N$. Applying the Hoeffding bound, one has

$$p[|\hat{p}_{ik}^{t} - p_{ik}^{t}| \geq \epsilon] \leq 2 \exp(-\frac{2\epsilon^2 N^2}{N})$$

Let $p[|\hat{p}_{ik}^{t} - p_{ik}^{t}| \geq \epsilon] \leq \delta$, which gives us the lower bound of N: $\frac{1}{2\epsilon^2} \ln(\frac{2}{\delta})$. □

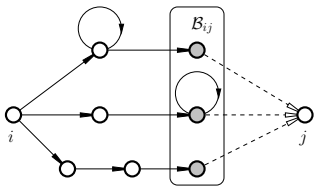

Figure 5: Pattern-guided hitting time estimation.

We then examine how the error of hitting probability estimation impacts the estimation of the hitting time h_{ij}^T. Assume that the error $|\hat{p}_{ik}^t - p_{ik}^t|$ is bounded by ϵ. From Equation 7, it can be derived that the maximum influence occurs when h_{ij}^t for all $t \in [1, \overline{T}]$ is decreased or increased by ϵ. We have the following derivation:

$$\begin{aligned} |\hat{h}_{ij}^T - h_{ij}^T| &\leq -\sum_{t=1}^{\overline{T}} t \cdot \epsilon + \epsilon \cdot \overline{T}^2 + \epsilon \cdot \overline{T} \cdot \overline{T} \\ &= [(T - \frac{1}{2}) \cdot \overline{T} - \frac{\overline{T}^2}{2}] \cdot \epsilon \end{aligned}$$

It can be derived that this error bound is monotonically increasing as $\overline{T}$ varies within the range of $[0, T-1]$. It also reveals a trade-off between the storage (maintenance) cost and the estimation accuracy of hitting time: at one extreme of the spectrum, if one caches the hitting time for all nodes with hitting times below T to every $j \in \mathcal{V}_S$ ($\overline{T} = 0$), the estimation error is minimized; at the other extreme of the spectrum, if one completely relies on on-line estimation ($\overline{T} = T - 1$), the storage cost is minimized, while the estimation error also reaches its maximum. The optimal setting of $\overline{T}$ will be discussed in Section **??**.

Now, we are ready to introduce the construction of D-operator, which is composed by two steps: 1) apply S-operator to obtain the estimation of p_{ik}^t for every $k \in \mathcal{N}(j, \overline{T})$ and $t \in [1, \overline{T}]$; 2) apply Equation 7 to the estimated probabilities and cached results to estimate h_{ij}^T.

4.3 Pattern-Guided Estimation

It is noted that D-operator developed in Section 4.2 features fairly low computational complexity; for large graphs, however, one may expect even more efficient solutions that could roughly tell us if the information leakage between a source and a sink is above certain threshold, based on certain local information only. Here, we introduce local-pattern-based estimation which leverages the local graph patterns of the source and sink, and gives quick estimation of the hitting time.

Intuitively, for given source i and sink j, we intend to find a set of "hot spots" $\mathcal{B}_{ij}$ such that a walk from i tends to go through each node $k \in \mathcal{B}_{ij}$ with high probability; we then base the estimation of h_{ij}^T on this set of hot-spot nodes. This scenario is illustrated in Figure 5: three walks (all of length 3) hit the three nodes in $\mathcal{B}_{ij}$, respectively.

More formally, let $\mathcal{W}$ be a set of walks starting from i, all of length of $\overline{T}$ (allowed to contain self-loops). Assume that $w \in \mathcal{W}$ sequentially consists of the nodes $w = (n_0^w, n_1^w, \ldots, n_{\overline{T}}^w)$ where $n_0^w = i$. The occurrence probability of w, $p(w)$, is defined as: $p(w) = \prod_{k=1}^{\overline{T}} p_{n_{k-1}^w n_k^w}$. Let $e(w)$ be the last node of w. We have the following bounds, derived from Equation (7):

$$h_{ij}^T \leq \sum_{w \in \mathcal{W}} p(w) \cdot (\overline{T} + h_{e(w)j}^{\overline{T}}) + (1 - \sum_{w \in \mathcal{W}} p(w)) \cdot T \tag{8}$$

Note that for clarity, we assume that none of the walks hit j; while it is fairly easy to remove this assumption following the formation of Equation (7).

A moment of reflection shows that the sampling procedure of D-operator essentially attempts to enumerate all such possible walks and estimate their occurrence probabilities according to the numbers of their

appearances. To ameliorate the complexity of enumerating all possible walks, we intend to pick a set of walks $\mathcal{W}$ such that every walk $w \in \mathcal{W}$ features high occurrence probability $p(w)$, i.e., $e(w)$ is a hot spot, then we can obtain fairly tight estimation regarding h_{ij}^T without strictly following the construction of D-operator.

We introduce a learning-based framework: 1) one first learns a set of *walk patterns* that feature high occurrence probability and high appearance frequency in the network; 2) one then caches such patterns in a fast lookup table structure; (3) on selecting the walks for a given source, one leverages the table as the guide to efficiently find high-probability walks. We first introduce the concept of walk pattern.

Definition 9 (Walk Pattern). *A walk pattern p is a sequence of relationship types $(r_1^p, r_2^p, \ldots, r_{L_p}^p)$, where L_p $(L_p \leq \overrightarrow{T})$ is its length. The occurrence probability of p, $p(p)$, must be above a predefined threshold. A walk $w = (n_0^w, n_1^w, \ldots, n_{L_p}^w)$ is an instance of p, if $r_{n_{i-1}^w n_i^w} = r_i^p$ for all $i \in [1, L_p]$. The frequency of p is the number of its instances in the network, which must be above a minimum support.*

Note that a walk pattern does not include any self-loops; after obtaining a walk as guided by a pattern, one needs to populate the walk to satisfy the expected length $(\overrightarrow{T})$ by incorporating self-loops.

Discovering frequent substructures in graphs has been a prominent topic in graph mining, with a plethora of tools available [11, 15, 23]; instead of re-inventing the wheels, here, we focus our discussion on how to leverage the discovered walk patterns to perform fast estimation.

Like a set of strings, a set of walk patterns can be organized in a trie, with each edge representing a relationship type. It is clear that each path (a set of edges) from the root to a (non-)leaf node represents a pattern. At each node, we store the occurrence probability of the corresponding pattern. For a given source i, following a breath-first search paradigm, we match the trie with the sub-network rooted at i. If a path from i to k, $(n_0, n_1, \ldots, n_{L_p})$, matches a walk pattern p. Let $p(n_e)$ $(e \in [0, L_p])$ denote the probability of self-loop at node n_e. We increase the hitting probability $hp_{ik}^{\overrightarrow{T}}$ by the following quantity $\binom{\overrightarrow{T}-1}{L_p-1} \cdot p(p) \cdot (\min_e p(n_e))^{(\overrightarrow{T}-L_p)}$. Essentially, here we populate the pattern with $(\overrightarrow{T} - L_p)$ self-loops to satisfy the expected length $\overrightarrow{T}$, and count all possible ways of adding self-loops as $\binom{\overrightarrow{T}-1}{L_p-1}$. This formation provides a lower bound for the sum of probabilities that the instances of p hit k. After enumerating all the matched patterns, we estimate h_{ij}^T based on Equation (8). We refer to this procedure as P-operator.

```
Input: source subject i, walk pattern trie T
Output: a walk with high occurrence probabilities
// initialization
w ← ∅, n ← i, t ← root of T;
// ↔ denotes correspondence
while |w| < T⃗ do
    if n ↔ t in T then
        // pattern guided sampling
        // R_t denotes descendent relationships of t
        for each n⁺ ∈ N_n do
            if r_{nn⁺} appears in R_t then adjust p_{nn⁺};
        normalize outgoing probabilities {p_{n·}} if necessary;
    sample n⁺ among N_n according to p_{nn⁺};
    append n⁺ to w;
    // update current nodes
    if n ↔ t and n⁺ ↔ t⁺, a descendent of t then
        t ← t⁺;
    n ← n⁺;
output w;
```

Algorithm 5: P-operator (pattern-guided sampling).

The pattern-guided sampling procedure, which we referred to as the P-operator, is sketched in Algorithm 5. It differentiates itself from the regular S-operator in that as walking in the network, it also descends down the walk-pattern trie if possible (i.e., the walk so far follows certain pattern in the trie). On selecting (sampling) an outgoing link, it favors those with relationship types appearing in the outgoing links of the corresponding node in the trie, by adjusting their weights.

5 Implementation of Library

In this section, we show how to implement the library of operations based on the set of atom operators introduced in Section 4. We are not arguing that this library is comprehensive; however, it is readily extensible: new operations can be constructed by composing the atom operators.

operator	input	output
J (join)	source i, sink(s) $j(\mathcal{J})$	residual information of i at $j(\mathcal{J})$
M (merge)	sink j	sources $\mathcal{I}$ with high leakage to j
D (diffuse)	source i	sinks $\mathcal{J}$ with high leakage from i
X (cross)	subjects $\mathcal{J}$	objects $\mathcal{I}$ accessible to $\mathcal{J}$
A (alarm)	subjects $\mathcal{J}$	alarms associated with $\mathcal{J}$
$\overline{\text{R}}$ (refer)	object j	objects $\mathcal{I}$ depending (transitively) on j
$\underline{\text{R}}$ (refer)	object j	objects $\mathcal{I}$ depended (transitively) by j

Table 1: List of atom operators.

Following, we show how to compose these atom operators to form high-level operations. Using these atom operators, we propose an expressive algebra framework. We use i, j, k to denote individual node, $\mathcal{I}$,J,K to represent sets of nodes. In addition to the atom operators introduced in Section 4, we further introduce three simple operators, X-operator (cross), which returns accessible objects for given subjects following inter-network links, R-operator (refer), which returns the referred (or referring) objects by a given object, and A-operator (alarm), which returns the set of alarms associated with a set of subjects. Also, we use $\otimes$ to denote the composition of two operators, and $\odot$ to denote an iterator which iterates over the set of elements. The list of operators is shown in Table 1.

Operation 1: Prior-Flow Estimation

$$\texttt{J}(i, \texttt{X} \otimes \texttt{M}(j))$$

For given object i and subject j, the operation of *prior-flow* estimation determines the potential information leakage existing in the current socio-information network before an access request $(j \to i)$ is granted. The estimation is implemented by composing J-, X- and M-operator. (1) One first applies M-operator to determine the set of source subjects κ_s featuring high information leakage measures to the sink subject j. (2) One then applies X-operator over κ_s to find the set of objects κ_o accessible to κ_s. (3) Taking κ_o and i as input, J-operator estimates the residual information of i at κ_o. If the residual information is above certain threshold, the flow is considered as informative enough for j to learn i via the network.

Operation 2: Posterior-Flow Estimation

$$\texttt{J}(i, \texttt{X} \otimes \texttt{M}(\odot\texttt{D}(j)))$$

For a given access request $(j \to i)$, the operation of *posterior-flow* estimation identifies the set of subjects who possess potential access to i because of j. This operation is implemented in the following steps. (1) One first applies D-operator to j to identify the set of sink subjects κ_s featuring high leakage measures from j. (2) For each $k \in \kappa_s$, one follows the procedure of prior-flow estimation (without the inter-network link $\overline{ij}$) to estimate the flow between i and k. (3) One then removes those whose prior-flow estimations are above certain threshold; that is, they have potential access to i even without j's help.

Further, one might intend to evaluate if the grant of access will trigger any *alarm*, a violation of classification/clearance rule. Intuitively, if the posterior-flow estimation indicates that a subject $k \in \kappa_s$ now has potential access to i, while $\texttt{class}(i) > \texttt{clear}(k)$, an alarm is raised.

The second set of operations are designed to support updating social and information networks. In particular, we focus our discussion on *incremental* update, e.g., new subjects/objects are added, new social relationships are created, etc., and similar discussion applies to *decremental* update.

Operation 3: Adding A New Object

$$\mathtt{J}(\odot\underline{\mathtt{R}}(j), j)$$

It is noted that each object may involve in multiple viewpoint information networks depending on its dependency on other objects, e.g., one blog refers to multiple blogs; hence, on inserting a new object, one needs to update all these affected networks. Here, we assume that the objects are inserted according to their orders of dependency; that is, an object can be inserted only after all its dependent objects have been inserted.

(1) Let j be the object to be inserted, one first applies $\underline{\mathtt{R}}$-operator over j to identify all objects $\mathcal{K}_o$ directly or indirectly depended by j (closure). (2) For each object $i \in \mathcal{K}_o$, one applies J-operator to estimate the residual information of i at j. (3) Meanwhile, one creates a new viewpoint network with j as the single node (root). Note that this operation affects the existing information flow between objects and subjects only through adding new access upon j, which is implemented mainly by Operation 2.

Operation 4: Updating A Link in Information Network

$$\mathtt{J}(i, \odot\overline{\mathtt{R}}(j))$$

This operation updates the degradation rate on a link $\overrightarrow{\iota_{ij}}$ of a viewpoint information network $\mathcal{G}_I^i$. Clearly, this will affect the estimation of residual information of i at all the objects dependent on this link. Starting from j, one incrementally applies J-operator to update the estimation of residual information at affected objects.

Operation 5: Adding A New Subject

$$\mathtt{A}(\mathtt{D}(j))$$

This operation differs from that of adding a new object in that in social network, information flows are estimated dynamically; instead, we focus on evaluating if any access-control alarms are triggered. Specifically, an alarm is an object-subject pair, which specifies that the subject *must* not access the object.

(1) One first applies D-operator over the to-be-inserted subject j to identify the set of subjects $\mathcal{K}_s$ with high leakage measures from j. (2) One applies A-operator over $\mathcal{K}_s$ which returns a set of alarms $\mathcal{A}$. (3) One further evaluates if any alarm in $\mathcal{A}$ is triggered by following the procedure of prior-flow estimation.

Operation 6: Updating A Relationship in Social Network

This operation updates an existing relationship or adds in a new relationship in the existing social network. Clearly, updating a link could potentially affect the estimation of information leakage for subjects within T-step walk distance. The implementation of this operation is similar to that of Operation 5. The details are omitted here.

6 Exploiting Network Evolution

So far, we have been focusing our discussion on estimating information flows for static snapshots of information and social networks; while the actual networks may evolve over time, which makes it imperative to take account of such dynamics in risk estimation. Typically, the evolution of information network is much less evident, compared with that of social network; therefore, we simply apply Operation 3 and 4 introduced in Section 5 to accommodate the update of information network, and concentrate on the impact of social network evolution [16, 17]. In particular, in this paper, we focus on social link creation, given its much higher frequency than other types of evolution.

To take account of the impact of network evolution, one needs to have the prediction model for the link creation process. While studying the concrete prediction model is beyond the scope of this paper; in our experiments, we adopt the model developed in [16], which focuses on predicting "triangle-closing" links that

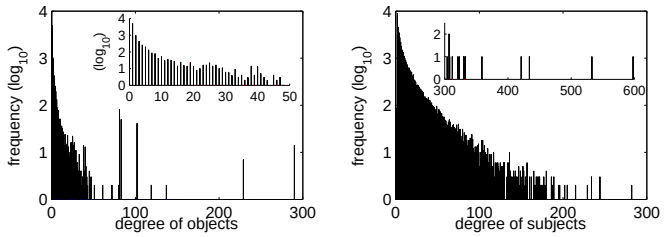

Figure 6: Degree distributions in information and social networks.

connect two-hop-away nodes. A missing mosaic that is imperative for our purpose is the determination of relationship type for the newly created link. We intend to determine the relationship type based on the types of the two links which it "shortcuts". Specifically, we intend to estimate the probability that the link $\overline{ik}$ takes on type r ($r \in \mathcal{R}$) given the types of links $\overline{ij}$ and $\overline{jk}$ as r_{ij} and r_{jk}, respectively, $p(r_{ik} = r | r_{ij}, r_{jk})$ ($r \in \mathcal{R}$).

In current implementation, we assume that $p(r_{ik}|r_{ij}, r_{jk})$ follows a Multinomial distribution, with a Dirichlet prior. Specifically, consider the social network $\mathcal{G}_S$ at a specific point. Among triangles with two neighboring (directed) links as r_{ij} and r_{jk}, let α_r denote the number of triangles with the rest link as r ($r \in \mathcal{R}$). The parameters μ of the Multinomial distribution is determined by the following Dirichlet distribution:

$$p(\mu|\mathcal{G}_S) = \frac{\Gamma(\alpha_0)}{\prod_r \Gamma(\alpha_r)} \prod_r \mu_r^{\alpha_r - 1}$$

where $\alpha_0 = \sum_r \alpha_r$ and μ_r corresponds to type r in the Multinomial distribution. As the network $\mathcal{G}_S$ evolves, one updates α accordingly.

With the help of this prediction model, we can now incorporate network evolution in estimating potential information leakage. Specifically, when performing an atom operator at time-stamp t_0, in addition to the current materialized social network, we also consider all implicit links that could be potentially created in a future time window W; the information leakage measure, however, is temporally discounted. One possible scheme could be: for a link that would be created at the time-stamp $(t_0 + t)$, its weight is discounted by a multiplier of $\exp(1 - \frac{W}{W-t})$.

7 Empirical Evaluation

This section presents an empirical study of our network-centric access control model. The experiments are specifically designed with the following three objects: 1) we intend to measure the potential risk of information leakage incurred by ignoring the networking relationships among social subjects and information objects; 2) we aim at evaluating the execution efficacy of the library of network-centric access control operations; 3) finally, we attempt to show the impact of network evolution over access control risks and how well our model responds to such network dynamics. We start with describing the datasets and the setup of the experiments.

7.1 Experimental Setting

Our experiments used three datasets collected from real-life social and information networks.

The first dataset is an archive of *tweet* messages collected from the Twitter site, over the period of October and November, 2009, which contains 18,617,827 tweets, involving 203,222 users.

The second dataset is the social network corresponding to a subset of IBM employees who participated in the SmallBlue project [18]. The dataset consists of two snapshots of the social network as of January 2009 and July 2009, involving 41,702 and 43,041 individuals, respectively. The personal information regarding

Attribute	Description
email	email address of user `s` (identifier of subject)
url	url `o` bookmarked by `s` (identifier of object)
tags	bookmark tags made by `s` regarding `o`
time	time-stamp that `s` accesses `o`

Table 2: Attributes and descriptions.

each individual includes his/her (i) work location, (ii) managerial position, and (iii) social connections with other employees. We consider that such information in combination encodes the social relationships among individuals. Specifically, for two individuals sharing a social connection, we define their relationship type based on (i) if they share a same work location and (ii) their managerial relationship. The degree distribution of the social network is shown in the right plot of Figure 6. It is noticed that the distribution follows a power law-like function fairly strictly.

The third dataset is an archive of bookmarks tagged by the individuals appearing in the first dataset, collected by DOGEAR [3], a personal bookmark management application that as well supports sharing the community's bookmarks. The archive contains 20,870 bookmark records, relevant to 7,819 urls. Attributes of interest to us are listed in Table 2; in particular, *email* and *url* uniquely identifies a user (a subject) and a webpage (an object), respectively, and *tags* encode the semantics of the object. Let $\mathcal{T}_i$ be the collection of tags suggested by users regarding i. We consider the set of objects $\mathcal{O}_i = \{j | \mathcal{T}_j \cap \mathcal{T}_i \neq \emptyset\}$ as the objects depending on i. For a given object i, we construct its viewpoint information network $\mathcal{G}_I^i$ as a multi-layered network, the objects of each layer corresponding to a *maximum possible* cover of $\mathcal{T}_i$, and the objects are selected in a greedy manner to maximize the current cover. There is a directed link between two objects k and j at two adjacent layers if $\mathcal{T}_k \cap \mathcal{T}_j \neq \emptyset$, and the weight w_{kj} is defined as $w_{kj} = |\mathcal{T}_k \cap \mathcal{T}_j| / |\mathcal{T}_k|$. The degree distribution in such viewpoint networks is shown in the left plot of Figure 6. Compared with social network, this distribution demonstrates more irregular patterns; certain high degrees appear with high frequencies.

All the core algorithms (the library of operations) are implemented in Java. The experiments are conducted on a workstation with 2.80GHz Intel Celeron CPU and 512 MB RAM, running Windows XP.

7.2 Experimental Results

Validity of Leakage Model

In the first set of experiments, we use the Twitter dataset to validate the covert flow model on real subject and object network platform. We set up the experiments as follows. On Twitter, the subject network is constructed according to the following/followed relationships: one user (subject) s_i opts to follow another user s_j if s_i wishes to receive messages (*tweets*) from s_j; also, s_i can "leak" (*re-tweet*) the tweets from s_j, which may be further leaked by followers of s_j. Clearly, such leakage can happen between two remotely connected subjects due to the network effects. We intend to apply the covert flow model to quantify the likelihood that the information (tweets) possessed by one subject leaks to another subject in the network, and compare the estimated covert flow with actually measured leakage (number of re-tweets). We use the data corresponding to October, 2009 to collect the overall statistics regarding each subject, particularly the number of received tweets and among them the number of re-tweets, which we use to set up the parameters $\{w\}$ and $\{c\}$ in Section **??**. We apply the model to predict the leakage likelihood for the period of November, 2009.

For a specific pair of users s_i and s_j, we consider the severity of leakage as the number of re-tweets s_j posts with original tweets from s_i during the period of November, 2009. For each observed leakage severity, Figure **??** shows the corresponding estimated covert flow (leakage likelihood) averaged over pairs of users demonstrating such severity. One can notice the high correlation between the estimated flow and the actual leakage severity, indicating that the covert flow model captures the essence of leakage patterns. Further, we look at individual level comparison of estimated covert flow and observed leakage severity. We randomly

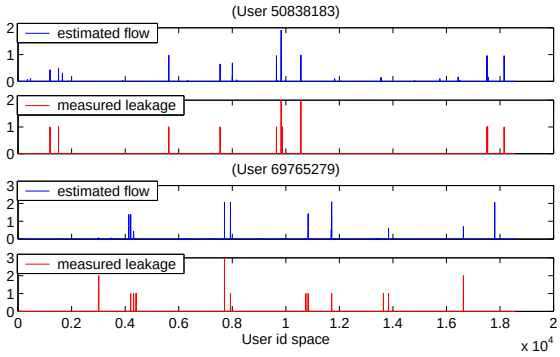

Figure 7: Individual level comparison of estimated covert flow and observed leakage severity.

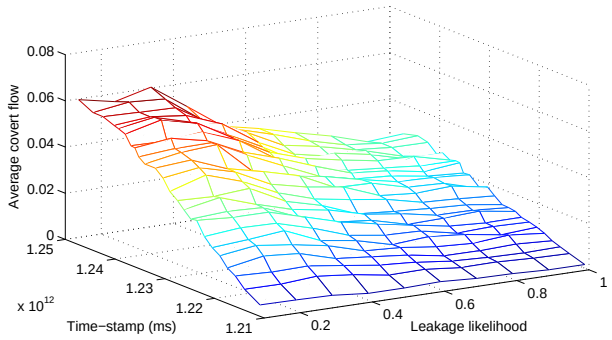

Figure 8: Covert information flows measured at different time-stamps.

pick two (sources) users, measure the leakage severities to the rest users, and compare the results with the suggested likelihood by our model. As shown in Figure 7, it is noticed that the predicted "peaks" match well with the actually measured results. Note that we are not attempting to model the actual leakage quantity (e.g., the number of re-tweets here); instead, our model strives to offer commensurate leakage indication which may be configured to fit application-dependent semantics. The concrete mechanism is beyond the scope of this paper.

Impact of Covert Flow

Next, we intend to evaluate the impact of covert flow existing in the social and information networks over the risk estimation in access control, more specifically, the risk of information leakage that would be underestimated if ignoring the network effects among subjects and objects.

We use the SmallBlue and Dogear datasets to construct the subject-object networks (details in Sec-

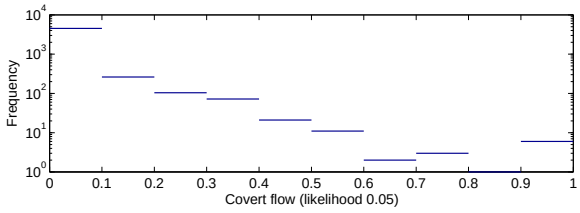

Figure 9: Distribution of covert flows of 5K randomly generated requests (leakage likelihood 0.5).

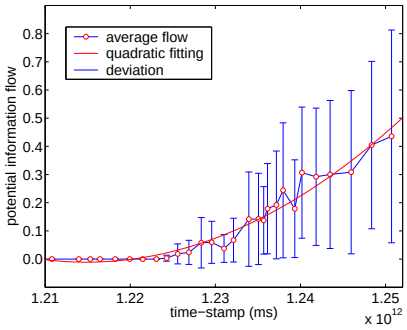

Figure 10: Potential information flows measured at different time-stamps.

tion **??**). We consider an bookmarking action as an access; hence, each access request q is associated with a time-stamp, t_q. At each specific time-stamp t^*, we assume that the set of requests with time-stamp smaller than t^*, $\{q|t_q \leq t^*\}$, have been granted. At each t^*, we randomly generate a set of access requests, and evaluate the covert flows with respect to these requests.

More concretely, we consider the history from 12/01/2005 to 07/20/2009, 20,870 access requests in total. At a step of 1,000 access requests, we evaluate the covert flows for 5K randomly generated requests. The trend of average covert flow with respect to time-stamp and leakage likelihood is plotted in Figure 10. It is noticed that as more requests are granted, the average covert flow increases significantly. This is explained by that the newly-created inter-network links between object and subject networks generally increase the covert flow capacity between the two networks, which also implies the non-negligible impact of the network effects among subjects and objects over access control risk estimation.

We look further into the distributions of covert flows of subject-object pairs. For the time-stamp 07/20/2009, we measure the covert flows for 5K randomly generated requests. Figure 11 shows the result. The distribution demonstrates a long tail, which is attributed to the heterogeneity of subject and object networks; that is, there exist "hot" spots in both networks, which feature large covert flows; the existence of "hot" spots necessitate careful risk estimation before making access control decisions.

Risk of Potential Information Leakage

In this set of experiments, we intend to evaluate the impact of potential information flow existing in the social and information networks over the risk estimation in access control infrastructure, more specifically, the risk of information leakage that would be under-estimated if ignoring the networking relationships among social subjects and information objects.

Specifically, each access request $\mathtt{q}$ is associated with a time-stamp, $t_{\mathtt{q}}$. For each specific time-stamp t^*, we assume that the set of requests with time-stamp smaller than t^*, $\mathcal{Q}_{t^*} = \{\mathtt{q}|t_{\mathtt{q}} \leq t^*\}$, have been granted. At each t^*, we randomly generate a set of access requests; with respect to these requests, we evaluate the potential information flows (in the scale $[0, 1]$).

More concretely, we consider the access history from 12/01/2005 to 07/20/2009, 20,870 access requests in total. At a step of 1,000 access requests, 'we evaluate the potential information flows for 0.5k randomly generated requests. The trend of average information flow and its deviation are plotted in Figure 10. It is noticed that as more requests are granted, the average potential information flow increases significantly. This is explained by the fact that the newly-created inter-network links between information and social networks generally increase the flow capacity between the two networks, which also demonstrates the non-negligible impact of the networking relationships among information and social networks over access-control risk estimation. It is also interesting to notice the increasing deviation of potential information flow, which is attributed to the heterogeneity of social and information networks. That is, there appear "hot" spots in both networks, which feature large incoming or outgoing information flows; while, correspondingly, "cold"

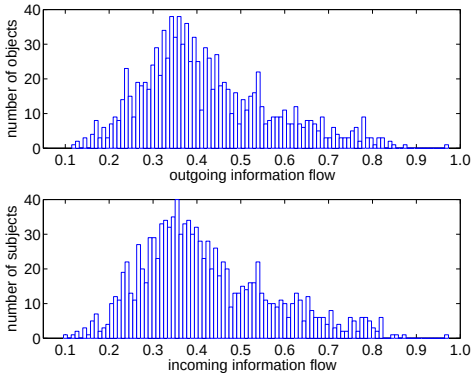

Figure 11: Distributions of outgoing information flows (for objects) and incoming information flows (for subjects).

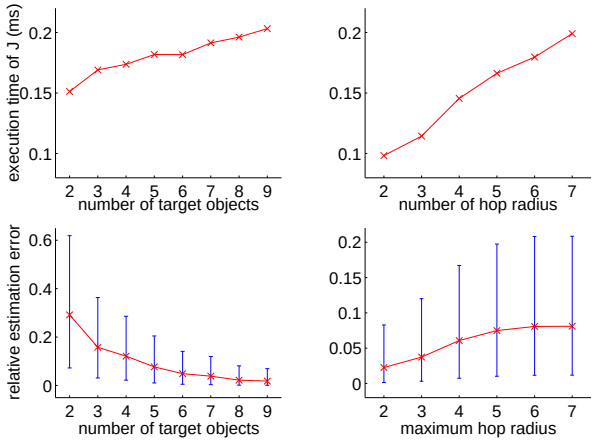

Figure 12: Average execution time and relative estimation error per operation for varying maximum hop radius and number of objects. Default parameter setting: maximum hop radius 6, number of target objects 5.

spots feature insignificant flows.

To validate our conjecture, we look further into the distributions of outgoing (for objects) and incoming (for subjects) information flows. In specific, for the time-stamp 07/20/2009, we evaluate the potential information flows for 2,000 randomly generated access requests, and calculate the average information flows for the involved objects and subjects. Figure 11 shows the distributions of outgoing and incoming information flows. It is clear that both distributions feature long tails, and the majority of the weights concentrate around the flow rate of 0.35. The existence of "hot" spots necessitate careful risk estimation before making access-control decisions.

Efficacy of Risk Estimation Operations

In the second set of experiments, we are interested in measuring the efficacy of the proposed risk-estiamtion operations. We intend to study how these operations scale with (1) the network complexity (in terms of intra-network and inter-network links), (2) the required estimation accuracy, and (3) the trade-off between space and time complexities.

Each operation in the library is constructed by composing a set of atom operators; meanwhile, the three

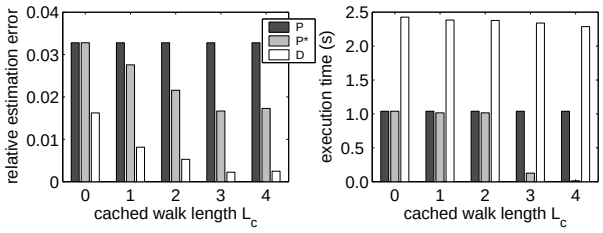

Figure 13: Average execution time and relative estimation error of D-operator and P-operator under varying cached walk length L_c. Default setting: $L_m = 5$, $N = 1000$, minimum support = 0.05.

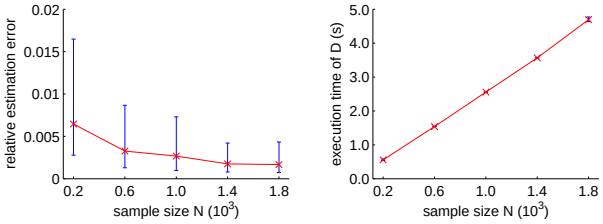

Figure 14: Average execution time and relative estimation error of D-operator under varying sample size N ($L_c = 3$).

factors above influence each atom operator in significantly different manners, making it fairly difficult to directly characterize their overall impact over the scalability of each operation. Hence, instead of attempting to directly study the efficacy of each operation, we focus our discussion on the level of atom operators. Due to the space limitation, in the set of atom operators {J, E, M, D, P}, we particularly concentrate on J-, D- and P-operator, given the more complicated trade-offs involved in their implementation.

We start with J-operator, which estimates the upper bound of residual information regarding a source object i in a set of target objects $\mathcal{J}$. We measure the scalability of J-operator along two orthogonal dimensions: (1) the cardinality of $\mathcal{J}$, i.e., the number of targets, and (2) the maximum hops of $\mathcal{J}$ to the source i. In the first case, for a given i, we fix the maximum hops (6 in our current implementation), randomly generate certain number ($|\mathcal{J}|$) of sinks, and perform J-operator. In the second case, for a given i, we fix the cardinality of J (5 in our current implementation), but vary the maximum hops.

The result is shown in Figure 12. It is clear that the average execution time of J-operator is an increasing function of both the number of target objects and maximum hop radius and ; however, the growth rates in both cases are non-significant. For example, as the maximum hop radius grows from 2 to 7, which virtually includes the whole information network, the average execution time only increases from 0.1 to 0.2 ms.

In both cases, we also measure the relative estimation error of J-operator. Given a source object i and a set of target objects $\mathcal{J}$, let $\mathcal{T}_{\mathcal{J}} = \cup_{j \epsilon \mathcal{J}} \mathcal{T}_j$ denote the set of tags associated with $\mathcal{J}$. We use $r = \frac{|\mathcal{T}_{\mathcal{J}} \cap \mathcal{T}_i|}{|\mathcal{T}_i|}$ as the exact residual information of i existing in $\mathcal{J}$. Let r' be the estimated upper bound of r. The relative estimation error is measured by $(r' - r)/r$. The bottom row of Figure 12 shows the relative estimation error as functions of the number of target objects and maximum hop radius. It is noticed that the relative error is a decreasing function of the cardinality of target objects, and the deviation shrinks as well. This is explained by the fact that the set of target objects is selected at random, a larger cardinality implies more residual information, i.e., the estimated upper bound appoximates the exact value better. Meanwhile, as expected, the estimation upper bound tends to become looser as the maximum hop radius increases, though with non-significant growth rate.

Given a social network and a given i, D-operator and P-operator essentially evaluate the i-th row in the hitting-time matrix H^T, $H^T_{i\cdot}$. D-operator is constructed by composing a random sampling scheme and the cached information generated by M-operator; while P-operator leverages the previous learned walk patterns

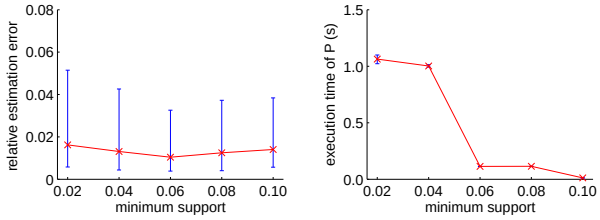

Figure 15: Average execution time and relative estimation error of P-operator under varying minimum support ($L_c = 3$).

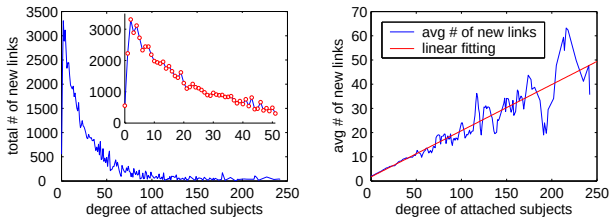

Figure 16: Total and average number of new relationships with respect to the degrees of their connected subjects.

to simulate the sampling procedure. In this set of experiments, we intend to evaluate the impact of the following factors over the execution efficiency and estimation accuracy of D-operator: (1) for fixed maximum walk length L_m, the cached walk length L_c (by M-operator), (2) the required estimation accuracy (ϵ, δ in Theorem 3), which is transformed to the sample size N, and (3) the minimum support for the learned walk patterns (for P-operator). The default setting of the parameters is: L_m = 5, ϵ = 0.02, δ = 0.2, L_c = 3, and minimum support = 0.05.

The impact of L_c is shown in Figure 13, where P, P* and D represent P-operator only, P-operator plus cached information, and D-operator, respectively. As expected, for fixed walk length, the cached intermediate result (L_c) significantly boosts up the estimation accuracy for both D and P. For example, for D-operator, when L_c reaches 4, the estimation error is almost non-noticable. Meanwhile, its impact over the execution time is less significant for D-operator than that for P-operator. This is explained by the fact that in D, each walk is executed completely until it hits the target or the maximum walk length is reached, while in P, the search stops once it hits a cached node. One can also notice that P-operator achieves estimation accuracy comparable to that by D-operator, but with considerable saving on execution time.

Further, we study the influence of sample size N over the performance of D-operator. The sampling operation dominates the complexity, which is demonstrated in Figure 14. Also, the sample size has non-neglectable impact over estimation accuracy. We observed that in most cases, $N = 2 \sim 3$k provides sufficient sample size. The impact of minimum support over P-operator is shown in Figure 15. It is interesting to notice that the estimation error demonstrates a "V" shape as the minimum support varies from 0.02 to 0.1. It is explained by that when the support is extremely low, a large number of trivial patterns are preserved, resulting in too many low-quality matches; while the support is overly high, only few extremely frequent patterns are maintained, leading to a number of missing matches.

Incorporation of Network Evolution

One critical feature that makes our access control paradigm useful is its capability of incorporating predicted network evolution in current risk estimation procedure. While measuring the accuracy of prediction model is orthogonal to this paper, here, we measure the tolerance of our model against prediction error; that is, given reasonable prediction accuracy, how well can our model produce accurate estimation in terms of overall

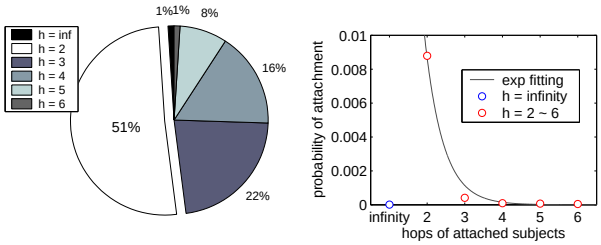

Figure 17: Fractions of new relationships with respect to the hops of their connected subjects and the attachment probabilities, where ∞ indicates that two subjects are disconnected in the original network.

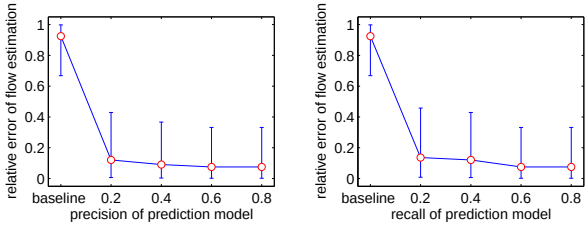

Figure 18: Relative estimation error with respect to the precision and recall of 2-hop link prediction model, where no prediction is employed in the baseline approach.

information flow.

Specifically, we take two snapshots of the social network as of January 2009 and July 2009, and consider the set of individuals appearing in both snapshots as the pool of subjects, which contains 32,028 individuals. From January 2009 to July 2009, 81,592 new relationships were created among these subjects. Figure 16 demonstrates the characteristics of these newly created links. It is noticed that 1) a majority of new relationships are attached to subjects with low degrees, which is possibly attributed to the power law-like distribution of degrees, and 2) the average number of new links per subject grows approximately linearly with respect to its current degree, i.e., "the rich get richer". Further, we study the locality of edge attachment with result shown in Figure 17. We measure the number of hops each new relationship spans, i.e., the length of the shortest path between the attached subjects. The left plot shows the fraction of new relationships corresponding to each hop distance h, and the right one normalizes this count by the total number of subjects at h hops, which counters the impact that the number of long-range neighbors grows exponentially. It is clear that in both plots, the fractions of new relationships decays exponentially in the hops, and a majority (over 51%) of new relationships span 2 hops, i.e., triangle-closing relationships.

Given such a large fraction of triangle-closing relationships, we assume a prediction model, e.g., [16], focusing on predicting 2-hop spanning relationships. We intend to study the robustness of our estimation model against the prediction error incurred by the prediction model. Specifically, for 0.5K randomly generated access requests, we measure the latent information flow over the snapshot social network as of July 2009 (exact flow), and compare it with that measured over the snapshot social network as of January 2009, in conjunction of the prediction model (estimated flow). We evaluate the relative estimation error as functions of the precision and recall of the prediction model, with default recall and precision set as 0.5. The result is shown in Figure 18. In both cases, the estimation made by baseline approach (without prediction) over the snapshot of January 2009 considerably deviates from that over the snapshot of July 2009; the relative estimation error reaches around 0.9. Employing the prediction model significantly improves the accuracy of information flow estimation; even with recall as 0.2, the average error is reduced around 0.2. However, as the precision (or recall) increase, the further accuracy improvement is slow; this is explained by that the model only considers 2-hop links, while other types of links account for 49% of new relationships.

8 Conclusion

In this paper we have described a novel network-centric access control paradigm that explicitly accounts for network-effects in information flow. We have shown that our approach is flexible and scalable by developing a suite of composable operators that can estimate prior and posterior information flow on social and information networks. We believe that examining access control models in terms of prior and posterior flows offers fundamentally new insights and opens up new possibilities in risk-based access control.

While our approach is rich and flexible, several challenges need to be addressed before it can be readily adopted. (1) Measures of information flow on each network edge (subject-subject and object-object). We have shown that several traditional access control models (such as MLS, RBAC, Chinese-wall like policies, etc.) may be encoded by suitably introducing 0/1 edge weights in our model; however, in order to meet our ambitious objective of risk estimation, it may be required to assign fractional edge weights (say, based on the rate of information diffusion between two subjects in the social network). While there may be several measures that can be used to derive such fractional weights, it is challenging to quantify such measures and to identify which of these measures are most effective. (2) Sensitivity to social network. We believe that our proposed approach is robust against small errors inherent in social network data; however, formal robustness analysis of our proposal is essential to quantify the efficacy of our approach when social/information network may be corrupted by an adversary. (3) Risk estimation usage. While our approach facilitates risk estimation, it may be quite challenging to use such risk estimates in performing access control decisions (e.g., do we use a threshold policy on risk estimates, if so, what is the right threshold; do we use a budget based policy that attempts to bound aggregate risk estimates, etc.).

References

[1] British spy chief's cover blown on Facebook: http://www. reuters.com/article/internetnews/idustre56403820090705.

[2] facebook - Press Room: http://www.facebook.com/press.

[3] Lotus Connections - Dogear: http://www.ibm.com/dogear.

[4] D. Aldous and J. A. Fill. Reversible markov chains, 1994.

[5] Anonymous. Network-centric access control - technical report, 2009.

[6] D. E. Bell and L. J. LaPadula. Secure computer system: unified exposition and multics interpretation. In *MITRE Corporation*, 1976.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., 2006.

[8] M. Brand. A random walks perspective on maximizing satisfaction and profit. In *SIAM'05: Proceedings of SIAM Conference on Optimization*, 2005.

[9] D. D. F. Brewer and D. M. J. Nash. The chinese wall security policy. *IEEE Symposium on Security and Privacy*, 0:206, 1989.

[10] P.-C. Cheng, P. Rohatgi, C. Keser, P. A. Karger, G. M. Wagner, and A. S. Reninger. Fuzzy multi-level security: An experiment on quantified risk-adaptive access control. In *SP '07: Proceedings of the 2007 IEEE Symposium on Security and Privacy*, 2007.

[11] L. Dehaspe, H. Toivonen, and R. D. King. Finding frequent substructures in chemical compounds. AAAI Press, 1998.

[12] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *J. ACM*, 19(2):248–264, 1972.

[13] D. Ferraiolo and R. Kuhn. Role-based access control. In *15th NIST-NCSC National Computer Security Conference*, 1992.

[14] K. Krukow, M. Nielsen, and V. Sassone. A logical framework for history-based access control and reputation systems. *J. Comput. Secur.*, 16(1):63–101, 2008.

[15] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, 2001.

[16] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008.

[17] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, 2003.

[18] C.-Y. Lin, N. Cao, S. X. Liu, S. Papadimitriou, J. Sun, and X. Yan. Smallblue: Social network analysis for expertise search and collective intelligence. In *ICDE '09: Proceedings of the 2009 IEEE International Conference on Data Engineering*, 2009.

[19] H. Qiu and E. R. Hancock. Image segmentation using commute times. In *BMVC'05: Proceedings of British Machine Vision Conference*, 2005.

[20] P. Sarkar, A. W. Moore, and A. Prakash. Fast incremental proximity search in large graphs. In *ICML '08: Proceedings of the 25th international conference on Machine learning*, 2008.

[21] M. Srivatsa, S. Balfe, K. G. Paterson, and P. Rohatgi. Trust management for secure information flows. In *CCS '08: Proceedings of the 15th ACM conference on Computer and communications security*, 2008.

[22] M. Srivatsa, P. Rohatgi, S. Balfe, and S. Reidt. Securing information flows: A metadata framework. In *QoISN'08: 1st IEEE Workshop on Quality of Information for Sensor Networks*, 2008.

[23] X. Yan and J. Han. Closegraph: mining closed frequent graph patterns. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2003.