Soc. Netw. Anal. Min. (2012) 2:39–52
DOI 10.1007/s13278-011-0032-x

ORIGINAL ARTICLE

Social positions and simulation relations

Joel Brynielsson · Lisa Kaati · Pontus Svenson

Received: 7 November 2010 / Revised: 22 May 2011 / Accepted: 27 May 2011 / Published online: 29 June 2011

Abstract Describing social positions and roles is an important topic within the social network analysis. Identifying social positions can be difficult when the target organization lacks a formal structure or is partially hidden. One approach is to compute a suitable equivalence relation on the nodes of the target network. Several different equivalence relations can be used, all depending on what kind of social positions that are of interest.One relation that is often used for this purpose is *regular equivalence*, or *bisimulation*, as it is known within the field of computer science. In this paper we consider a relation from computer science called *simulation relation*. The simulation relation creates a partial order on the set of actors in a network and we can use this order to identify actors that have characteristic properties. The simulation relation can also be used to compute *simulation equivalence* which is a related but less restrictive equivalence relation than regular equivalence that is still computable in polynomial time.We tentatively term the equivalence classes determined by simulation equivalence *social positions*. Which equivalence relation that is interesting to consider depends on the problem at hand. We argue that it is necessary to consider several different equivalence relations for a given network, in order to understand it completely. This paper primarily considers weighted directed networks and we present definitions of both weighted simulation equivalence and weighted regular equivalence. Weighted networks can be used to model a number of network domains, including information flow, trust propagation, and communication channels. Many of these domains have applications within homeland security and in the military, where one wants to survey and elicit key roles within an organization. After social positions have been calculated, they can be used to produce abstractions of the network—smaller versions that retain some of the most important characteristics.

Keywords Social network analysis · Social positions · Abstraction · Simulation equivalence

1 Introduction

Social network analysis (SNA) (Wasserman and Faust 1994; Scott 2000; Carrington et al. 2005) is a set of powerful techniques to identify social roles, important groups and hidden structures in organizations and groups. This methodology assumes that the ways the members of a group can and do communicate with each other are correlated with important properties of that group. There are a number of different measures that could be calculated for a given network and that tell us details about it. While SNA has a long and successful history within sociology, networks are everywhere in nature, and SNA and related methodology can be used to analyze a wide variety of problems (Newman et al. 2006).

One particular application that has gained interest lately is for military and security purposes: can we use SNA to find criminals or terrorists? While it is easier to use SNA in forensic analysis after a crime or terrorist act has been committed, it is sometimes still hoped that SNA in conjunction with other techniques could allow us to obtain early warnings about future incidents. Correlation of observed data about individuals, things, places, memberships, etc., may be used to detect organized crime or terrorist cells and networks through the observation of hidden relations and co-occurrences.

J. Brynielsson · L. Kaati (✉) · P. Svenson
Swedish Defence Research Agency, 164 90 Stockholm, Sweden
e-mail: lisa.kaati@foi.se

A network consists of a set of actors and a set of binary relations between the actors that describe their communication patterns. Most networks of interest are very large, making it difficult to visualize them. When analyzing large data sets, it is important to try to extract the most important features. This is particularly important in visualization. Displaying too much information to the user will distract them and lead to information overload instead of understanding the data. This can be remedied by finding a suitable *abstraction* of the data, so that less detail is displayed to the user, but the important characteristics of the data are still visible. For instance, in information fusion (Liggins et al. 2009), objects or events of interest are often clustered together and classified to form an aggregation. The challenge is to construct algorithms that cluster the data in such a way that the essential meaning of it is still retained in the new visualization. The goal of any abstraction is to transform a large network into a smaller one, so that the smaller is a useful summary of the original network.

Similar issues arise when analyzing networks, both social and other kinds. There are simply too many nodes and too many links to display. A traditional way of reducing the number of nodes and links shown is to cluster the nodes of the network, and only display the clusters and links between them. This is very similar to clustering objects using geospatial positions.

In order to facilitate easier visualization and analysis, we want to construct a smaller network that still has the most important characteristics of the original one. One way of doing this is to discover the community structure of the network, and only display these. Another is to look for actors in the network that are equivalent and construct a smaller network where all equivalent actors are merged.

In this paper, we follow the latter route and investigate how different equivalence relations from computer science can be used to detect social positions and roles in a social network. The standard view in SNA is that a social position refers to a collection of individuals who are similarly embedded in networks of relations (Wasserman and Faust 1994). A social position is therefore a property of actors in a network. For a given network a social position is defined as a subset of actors, namely those who have that property (Marx and Masuch 2003). Since positions are based on the similarities of ties among subsets of actors, actors that have the same position in a network do not need to be in direct (or indirect) contact with each other. Social roles are defined on the basis of social positions. A role is defined in terms of the patterns of relations between positions but there are also other possible definitions. For instance, in this paper we are interested in obtaining abstracted networks with a less restrictive equivalence relation than the standard equivalence relations (Wasserman and Faust 1994). An equivalence relation over the set of actors determines social positions in a network since the equivalence classes represent the different positions in a network. Using equivalence relations to create a smaller network is one way of creating an abstraction of the network. There are several different equivalence relations that can be used to describe positions in a network: automorphic equivalence, structural equivalence and regular equivalence are three of the most well-known equivalences in the literature (Wasserman and Faust 1994). Of these three relations, regular equivalence is the relation that is least restrictive. Computing these relations can be demanding; it is common to approximate the relations to get a better result.

In this paper we present a relation called simulation relation. The simulation relation creates a partial order on the set of actors in a network and we can use this order to identify actors that have characteristic properties.

The simulation relation can also be used to compute simulation equivalence. We use simulation equivalence to create equivalence classes that form positions in the network. The smaller network that is formed between these positions can be used as an abstraction of the larger network, and is easier to use for explorative visual analysis.

Finding such equivalence classes is interesting for many different applications. Our interest lies mainly in using the equivalence classes to produce abstractions of large networks that are easier for military intelligence analysts to study and visualize. An obvious application is for recommendation systems, where individuals who share the same social role might be expected to share the same taste. Another application is in enterprize incentive management, where company bonuses should be divided as fairly as possible among all employees. In a large corporation, employees who cannot be simulated (i.e., replaced) by other employees could be regarded as more valuable than others, and thus receive larger bonuses. The lack of redundancy could also be considered as a risk for the enterprize. In such cases, measures should be taken to make sure that a backup is created. In a similar way, the relation could be used to find weak points in other networks, for instance transport networks or information processing networks.

The techniques that we present in this paper can be used to identify social roles and social relations on actors in a network. An important motivation for defining different relations on the actors in a network is the fact that if we can define roles formally based on observed relations, we can detect emergent, unnamed roles and positions in social networks.

It is important to note that we do not argue that simulation equivalence is a "better" way of studying a social network than, e.g., regular equivalence. Both of these equivalence relations are useful, depending on the question one wishes to answer about the network. We discuss this issue further in Sect. 4 below.

1.1 Related work

There are several different methods to cluster actors in a social network into groups or cliques. Most clustering techniques are formed on the basis of social relations within a group. If we instead of clustering according to relations within a group want to use social positions or roles to cluster actors in a network, we can use the relations with other roles or positions in the network as a criteria for clustering actors (Freeman et al. 1992).

Community detection refers to an intuitively appealing clustering of the network nodes into "natural" subsets in the sense that the subsets "look like" reasonable communities with respect to the underlying relational patterns. By choosing an objective function that captures the desired intuition, sets of nodes that are related in the sense of the objective function can be extracted. However, specifying the most "realistic" formalization of a network community can be done in many more or less appropriate ways, and therefore the development of objective functions is a very active research area. Generally speaking, a "good" community is taken to refer to a subset of nodes that are (1) well connected among themselves, and (2) well separated from the rest of the graph. Hence, algorithms for network clustering are often based on a subjective quality measure that can be applied on a potential cluster. Therefore, in general, algorithms for network clustering differ in (1) how the quality of the proposed clusters is measured, and (2) what kind of optimization technique is used to obtain this desired quality. For a more in-depth discussion and description of some of the standard methods, see Kolaczyk (2009, pp. 102–111).

Formally described by Everett and Borgatti (1994) and put well into perspective by Wolfe (2011), regular equivalence is one way of clustering actors according to social positions. We use another closely related equivalence relation called simulation equivalence to cluster actors according to social positions. We consider both backward and forward simulation equivalence, both described by Lynch and Vaandrager (1995).

In the current work, we consider weighted directed social networks. As noted by Opsahl and Panzarasa (2009), most network measures do not take weights into consideration. However, it often makes sense to incorporate the extent/strength of relationships as weights to obtain a more accurate model of a social phenomenon. As exemplified by Fazeen et al. (2011), such weights can either be derived from available data (context-dependent weights), or be calculated based on more generic patterns (context-independent weights). Weighted networks are usually handled by either generalizing existing unweighted network algorithms to take weights into account, or developing completely new algorithms that do not have any connection to unweighted measures (Abdallah 2011). In this sense, our work provides both new unweighted clustering algorithms as well as generalizations of these algorithms to consider take weights. Regular equivalence for weighted networks is similar to the notion of bisimulation for weighted automata. Both backward and forward bisimulation for weighted automata are presented by Buchholz (2008).

1.2 Outline

In Sect. 2, an example aimed at providing an intuitive understanding of the mathematics is presented. In Sects. 3 and 4, the relevant definitions are given and the need for using several different equivalence relations when analyzing a network is motivated.

In Sects. 5 and 6, we present two examples in the form of communication scenarios that put the presented abstraction techniques into context, while Sect. 7 briefly describes military intelligence work and how it could benefit from abstraction techniques such as the one presented. The paper concludes with a summary and some suggestions for future work in Sect. 8.

2 Illustration of concepts

Analyzing equivalence classes is one way of making sense of the patterns of relations among actors in a network. The ability to define, theorize about, and analyze data in terms of equivalence is important since we want to be able to make generalizations about social behavior and social structure. In order to do this we think about actors as examples of categories. That is, sets of actors who are in some way defined as "equivalent."

To illustrate the difference between simulation equivalence and regular equivalence, we can use the network shown in Fig. 1. The network represents a group of terrorists and how they communicate with each other. Using simulation equivalence we obtain the equivalence classes {1, 2}, {3}, {4} and {5,6}. We have four different social

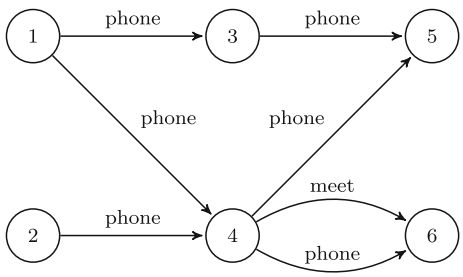


Fig. 1 A network representing the communication between a group of terrorists with different social positions

roles: a group of terrorist leaders (the group containing actors 1 and 2), two intermediate leaders that have different social positions, and a group of terrorists (the group containing actors 5 and 6).

If we use the more restrictive relation regular equivalence on the network we get the following equivalence classes: {1}, {2}, {3}, {4} and {5,6}. That is, we cannot capture that the two leaders (actors 1 and 2) have the same social position. This illustrates one of the uses of the simulation equivalence presented here. When analyzing very large networks, it would be beneficial to detect the similarity between actors such as 1 and 2 in the example, and computing equivalence classes based on simulation equivalence can help with this.

2.1 A preorder abstraction

Another use of the simulation relation is that it can be used to determine an even coarser abstraction. When computing simulation equivalence, a so-called preorder is obtained. This preorder divides the actors into a partial order and can be used to create an abstraction of a network. The abstraction is created by removing all actors that can be simulated by some other actor. In the network in Fig. 1, the actors 5 and 6 can be simulated by all other actors in the network (since they cannot do anything). Hence, they can be removed from the resulting abstraction. Actor 3 can be simulated by actor 4 and therefore we can replace actor 3 by 4. Actor 4 cannot be simulated by any other actor in the network. Actors 1 and 2 can be simulated by each other (that is, they are simulation equivalent). An abstraction of the network in Fig. 1 is shown in Fig. 2.

In this example, the abstraction of the network shows that there are two important groups in the network: the terrorist leaders and the intermediate leaders.

2.2 Weighted directed networks

In a more advanced approach we can add weights to the ties in the network.

Simulation equivalence is a less restrictive equivalence relation than regular equivalence, especially in the case with weights that are integers. Using regular equivalence, two states are considered equivalent only if they have similar ties *and* weights to other equivalence classes while in the case of simulation, one actor simulates another actor if the ties are similar and have *at least* the same weights (possibly more).

Regular equivalence for weighted networks is a quite restrictive relation since two actors are considered equivalent only if they have edges with similar weights to equivalent actors. In this example we use weights from the semiring of integers. This means that two edges can be combined using addition of their weights.

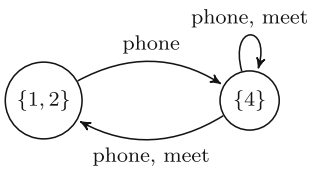


Fig. 2 An abstraction of the network in Fig. 1 using the simulation preorder

3 Definitions

In this section we present some definitions of networks, relations and equivalences.

3.1 Numbers and relations

Let S be a set, and let $\simeq$ and $\cong$ be binary relations on S, i.e., $\simeq$ and $\cong$ are both subsets of $S \times S$. We abbreviate the Cartesian product $S \times S$ by S^2. The relation $\cong$ is said to be *coarser* than $\simeq$ (or equivalently: $\simeq$ is a *refinement* of $\cong$), if $\simeq$ is a subset of $\cong$.

Let $\preceq$ be a binary relation on the set S. The relation $\preceq$ is a *preorder* (or *quasi-order*) if it is reflexive and transitive. It is an *equivalence relation* if it is a symmetric preorder. We write $\uparrow(s)$, where $s \in S$, for the set of elements $\{s' \mid s \preceq s'\}$ of elements in S that dominate s. We denote by $max_{\preceq}(S)$ the subset $\{s \mid \nexists s' \in S : s \preceq s' \wedge s' \not\preceq s\}$. That is, $max_{\preceq}(S)$ is the set of *maximal* elements of S with respect to $\preceq$. We denote by $=_{\preceq}$ the coarsest equivalence relation that is a subset of $\preceq$, i.e., $=_{\preceq}$ is equal to $\preceq \cap \preceq^{-1}$.

3.2 Example: preorder

Let $S = \{a, b, c\}$. The relation $\{(a, a), (b, b), (c, c), (a, b), (b, c), (a, c)\}$ is a preorder on S, but neither

$$\{(a,a),(a,b),(b,c),(a,c)\}$$

nor

$$\{(a,a),(b,b),(c,c),(a,b),(b,c)\}$$

are, as the first relation is not reflexive and the second is not transitive.

The *equivalence class* of an element $s \in S$ with respect to an equivalence relation $\simeq$ is the set $[s]_{\simeq} = \{s' \mid s \simeq s'\}$. Whenever $\simeq$ is obvious from the context, we simply write $[s]$ instead of $[s]_{\simeq}$. It should be clear that $[s]$ and $[s']$ are equal if s and s' are in relation $\simeq$, and disjoint otherwise. The equivalence relation $\simeq$ thus induces a partition $(S/\simeq) = \{[s] \mid s \in S\}$ of S.

3.3 Networks and graphs

An *alphabet* is a finite set of symbols. We write ε for the empty word.

Definition 1 (*Directed graph*) A *directed graph* (or network) is a tuple (V, E) where

- V is a finite set of vertices (or nodes), and
- $E \subseteq V \times V$ is a finite set of edges.

If G has vertices with labels in the alphabet Σ, then V can be partitioned into a Σ-indexed family $(V_a)_{a\in\Sigma}$ of sets of vertices. If G has edge labels in Σ, then E is a Σ-indexed family $(E_a)_{a\in\Sigma}$ of sets of edges, such that $E_a \subseteq V \times V$, for each $a \in \Sigma$. We thus only allow parallel edges if they are labeled with different symbols.

For each symbol $a \in \Sigma$, we denote by $E_a(u)$ the set of vertices $\{v \mid (u, v) \in E_a\}$ that can be reached from u along a-labeled edges. Similarly, for each $w \in \Sigma^*$, we have

$$E_w(u) = \begin{cases} \bigcup_{v\in E_a(u)} E_{w'}(v) & \text{if } w = aw', \text{for some } a \in \Sigma \\ & \text{and } w' \in \Sigma^*, \text{and} \\ \{u\} & \text{if } w = \epsilon. \end{cases}$$

Let Σ be an alphabet and $G = (V, E)$ a directed graph with edge-labels in Σ. The graph G has a *trace* $w \in \Sigma^*$ if there is a $v \in V$ such that $E_w(v) \neq \emptyset$. The *trace behavior* of a graph G is given by

$$L(G) = \bigcup_{v\in V} \{w \in \Sigma^* \mid E_w(v) \neq \emptyset\}.$$

3.4 Algebraic structures

Towards the end of Sect. 4, we recall a number of results concerning weighted simulation, and for this we need the following algebraic concepts.

A *monoid* is a set A together with a binary operation $\cdot$ from $A \times A$ to A and an element 1 in A that satisfy the following two axioms:

- The operation $\cdot$ is *associative* in that for every three elements a, b, and c in A, it holds that $(a \cdot b) \cdot c$ is equal to $a \cdot (b \cdot c)$.
- The element 1 is the *neutral* element with respect to $\cdot$; i.e., we have $a \cdot 1 = 1 \cdot a = a$, for all $a \in A$.

A monoid $(A, \cdot, 1)$ is *commutative* if $a \cdot b = b \cdot a$, for all a, b in A.

Moreover, a *semiring* is a tuple $(A, +, \cdot, 0, 1)$ where

- $(A, +, 0)$ is a commutative monoid and
- $(A, \cdot, 1)$ is a monoid.
- For every a, b, and c in A, it holds that $(a + b) \cdot c$ equals $(a \cdot c) + (b \cdot c)$ and $a \cdot (b + c)$ equals $(a \cdot b) + (a \cdot c)$, i.e. the multiplicative operation *distributes* over the additive.
- For every a in A, we have that $a \cdot 0 = 0 \cdot a = 0$. In other words, the element 0 is *absorptive*.
- Finally, 0 and 1 are distinct elements.

4 Simulation relations

Before we delve into the formal definitions, let us start with an intuitive description of simulation relation and simulation equivalence. A simulation on a graph G is an ordering (more precisely, a preorder) of the vertices of G, such that if there is an edge $v \to v'$ in G, then for every vertice u that dominates v, there is also an edge $u \to u'$ to some vertex u' that dominates v'. In other words, if a vertex u simulates a vertex v, then the freedom of movement at u is at least as great as it is at v. To put this into social network terms, we require that if v communicates with v', then all actors u that dominate v must communicate with some actor u' that dominates v'.

A maximal simulation equivalence is the coarsest equivalence relation contained in $\preceq$, i.e., $=_{\preceq}$. In a social network the equivalence classes correspond to different social roles.

The simulation preorder $\preceq$ can also be used to gain information about the actors in a network. The preorder contains information on the relations between the actors and if they simulate each other. A subnetwork containing only actors from the maximal vertices of $\preceq$ is an over approximation of the original network. This technique normally produces a small network but it does not preserve the trace behavior of the graph: although every trace that can be found in the original graph can also be found in the reduced graph, the converse is not true in general.

4.1 Relations

In this paper we consider the problem of finding the coarsest possible equivalence relation $\simeq$ on the set of vertices of a graph. We can use this information to collapse each equivalence class of $\simeq$ into a single vertex and produce a (hopefully) smaller graph.

Definition 2 (*Aggregated graph*) Let $G = (V, E)$ be a directed edge-labeled graph, and let $\simeq$ be an equivalence relation on the vertex-space V. The *reduced graph* $(G/\simeq)$ is the system $((V/\simeq), E')$ where

$$E' = \{([u]_{\simeq}, [u']_{\simeq}) \mid (u, u') \in E\}.$$

Note that E' is well-defined because $\simeq$ is an equivalence relation.

Definition 3 (*Regular equivalence*) An equivalence relation $\simeq \subseteq V \times V$ is a *regular equivlence* or *bisimulation* if $u \simeq v, a \in \Sigma$, and $v' \in E_a(v)$, implies that there is a $u' \in E_a(u)$ such that $u' \simeq v'$, and vice versa.

The coarsest regular equivalence can be computed in time $O(m \log n)$, where m is the number of edges and n the number of vertices of the input graph, using a divide-and-conquer technique developed by Hopcroft (1971) that was generalized to the nondeterministic case by Paige and Tarjan (1987). The major drawback with regular equivalence is that it is unnecessarily strict, and thus provides an unnecessarily weak reduction of the vertex space (Henzinger et al. 1995).

A less restrictive relation is *simulation preorder*. A vertex u *simulates* a vertex v if, for every symbol $a \in \Sigma$ and edge $(v, v') \in E_a$, there is an edge $(u, u') \in E_a$ such that u' simulates v'. Again translating to social network terms, this is equivalent to saying that u simulates v if for all actors v' such that v communicates with v' with the label a, there is always a link labelled a between u and an actor u' that simulates v' (recall that all actors simulate themselves).

Definition 4 (*Simulation preorder*) A preorder relation $\preceq \subseteq V \times V$ is a *simulation* if the fact that $v \preceq u, a \in \Sigma$, and $v' \in E_a(v)$, implies that there is a $u' \in E_a(u)$ such that $v' \preceq u$.

A pair of vertices $u, v \in V$ are *simulation equivalent* if $\preceq$ is a simulation and both $u \preceq v$ and $v \preceq u$ hold.

If the graph G under consideration has vertex-labels in the alphabet Σ, i.e. $V = (V_a)_{a \in \Sigma}$, then require that every simulation on V refines the partitioning $(V_a)_{a \in \Sigma}$.

A simulation preorder is in general not an equivalence relation. Instead, we can use the coarsest equivalence contained in $\preceq$, namely $=_{\preceq}$. Simulation equivalence $=_{\preceq}$ produces, in general, larger equivalence classes than regular equivalence.

If we are willing to settle for an over-approximation of the trace behavior, we can use the simulation preorder and create an abstraction of the graph. We first re-route all edges to and from each non-maximal vertex v to each of the vertices in $\uparrow(v)$. We then drop all vertices that are not in $max_{\preceq}(\preceq)$ together with the edges connected to them.

Definition 5 (*Simulation abstraction*) Let $G = (V, E)$ be a directed edge-labeled graph, and let $\preceq$ be a simulation relation on the vertex-space V. The *reduced graph* $(G/\preceq)$ with respect to $\preceq$ is the system (V', E'), and where $V' = (max_{\preceq}(V)/\preceq)$

$$E' = \{([u]_{\simeq}, [u']_{\simeq}) \mid u \in \uparrow(v) \cap max_{\preceq}(V) \wedge u' \in \uparrow(v') \cap max_{\preceq}(V) \wedge (v, v') \in E\}.$$

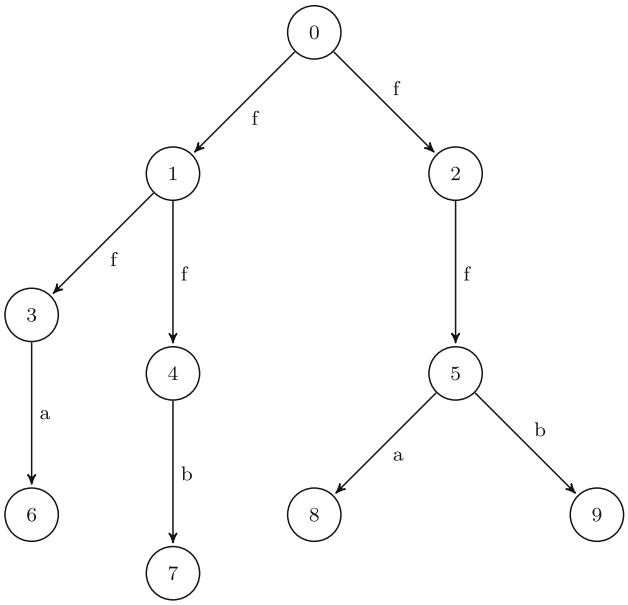


Fig. 3 The directed labeled graph G

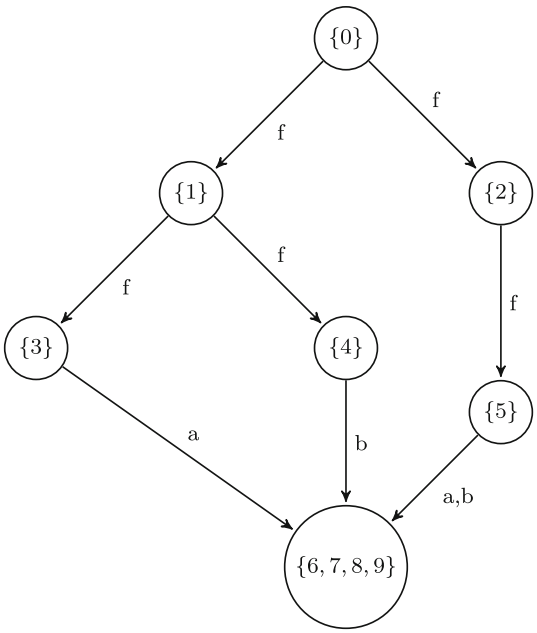


Fig. 4 The graph of Fig. 3 after computing regular equivalence

4.2 Example

Consider the directed labeled network G of Fig. 3, which we consider for illustration to be the organization chart of a company or terrorist group. Computing regular equivalence over G gives us the equivalence classes {0}, {1}, {2}, {3}, {4}, {5} and {6, 7, 8, 9}. The aggregated network is shown in Fig. 4. Using simulation equivalence we obtain the equivalence classes {0}, {1, 2}, {3}, {4}, {5 }, {6, 7, 8, 9},

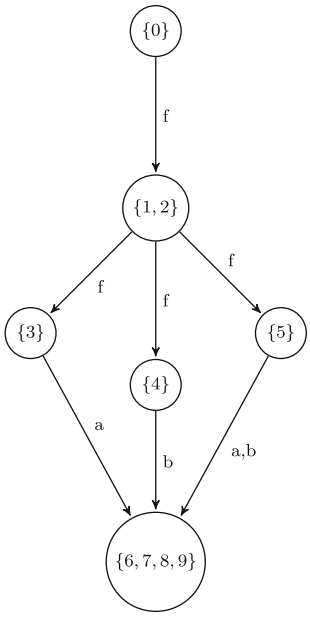
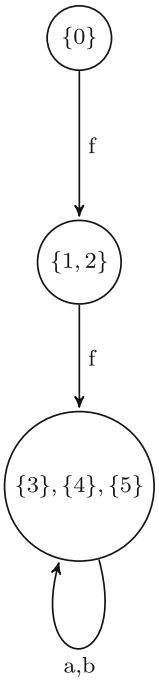


Fig. 5 The graph of Fig. 3 after computing simulation equivalence

Fig. 6 The graph of Fig. 3 after using simulation preorder abstraction. The nodes {6, 7, 8, 9} did not contain maximal vertices and was consequently dropped



the aggregated network is depicted in Fig. 5. Finally, using the simulation preorder abstraction, we obtain an aggregated network as depicted in Fig. 6.

By comparing Figs. 4, 5 and 6, we see that regular equivalence, simulation equivalence and preorder abstraction gives us three different views of the original network (Fig. 3). Regular equivalence essentially combines all leaf nodes in the tree of Fig. 3, enabling us to study the "middle management" parts of the depicted organization in detail. Simulation equivalence recognizes the similarity between nodes 1 and 2, and provides a simpler view of the same organization. Of course, the drawback with simulation equivalence is that we also lose some information as compared to both the original network and the regular equivalence view. Finally, Fig. 6 shows the coarsest equivalence relation introduced in this paper, the simulation pre-order. Here we have lost all structure of the original network except for the fact that the organization contains three layers of leaders.

It must be stressed that it is vitally important that the correct equivalence view is chosen for the specific question which the intelligence analyst is trying to answer. In this paper, we do not put forward simulation equivalence as a "better" or more useful way of analyzing a network than regular equivalence, but merely as a different way that, for some specific applications, could be used in conjunction with regular equivalence and other analysis methods to help analysts achieve increased understanding of the network under study.

4.3 Weighted relations

It is often useful to model quantitative as well as qualitative relations between vertices in a social network, such as trust, affection, or level of communication. The typical approach is to assign weights taken from some semiring A to the edges of the network, turning the edge relation into a (total) mapping that takes pairs of vertices into A.

Definition 6 A *weighted directed graph* or *weighted directed network* (over a semiring A) is a tuple $G = (V, E)$, where V is a finite set of vertices and $E: V \times V \to A$ is a mapping that assigns to every ordered pair of vertices in V a weight in A.

If G has edge labels in Σ, then we consider E to be a family $(E_a)_{a\in\Sigma}$ of mappings such that $E_a: V \times V \to A$, for every $a \in \Sigma$.

Regular equivalence for weighted directed networks is defined as follows.

Definition 7 (*Weighted regular equivalence*) An equivalence relation $\simeq \subseteq V \times V$ is a *regular equivalence* if $u \simeq v$ and $v' \in E(v)$, implies that there is a $u' \in E(u)$ such that $E(u, u') = E(v, v')$ and $u' \simeq v'$, and vice versa.

The definition of weighted simulation is similar. Definitions for simulation relations for *weighted tree automata* (since a string is a special kind of tree, the definitions can be directly applicable to graphs) are described by Maletti (2009).

Definition 8 (*Weighted simulation*) The preorder $\preceq$ is a *weighted simulation* if $v \preceq u$ implies that, for each $v' \in V$ and $a \in \Sigma$, there is an $u' \in V$ such that $E(u, u') \geq E(v, v')$ and $v' \preceq u'$.

As one could notice, the difference between regular equivalence and simulation equivalence is more observable in the weighted setting. This is because the weights of the edges have to be strictly equal to fulfil the requirements in the case of regular equivalence while in the case of weighted simulation, the weight of an edge has to be equal or greater than to satisfy the required conditions.

4.4 Backward simulation

Both regular equivalence and simulation equivalence define nodes as equivalent if they have equivalent successors. For these equivalences, the direction of the relation between nodes are important. If we instead of defining the equivalence relation based on successor define two nodes as equivalent if they have equivalent predecessor we get a different equivalence relation. This modification can easily be done by shifting the direction in the network and then running the same algorithm. Conceptually, this is the same as re-defining the meaning of the relations. For instance, in a network where ties are determined by e-mail communication and we have information that actor *a* sent an e-mail to actor *b*, we can choose arbitrarily to model this using either of the relations "sends e-mail to" or "receives e-mail from." This difference in semantics does not change the network as such in any way. However, depending on what question we ask about the network, we might get different results depending on what relation type we chose. We illustrate this for the simple network shown in Fig. 7.

If we interpret the relations in this network as meaning "sends an e-mail to," we might be interested in determining the simulation equivalences. The result is that the equivalence classes are $\{b, e\}$, $\{d, a\}$ and $\{c, f\}$. These equivalence relations tell us that there are e-mail senders, receivers and senders, and receivers in the network. Note that there is no distinction between *c* and *f*, even though they receive e-mails from different persons and *f* receives e-mail from somebody who also receives e-mails from other persons in the network.

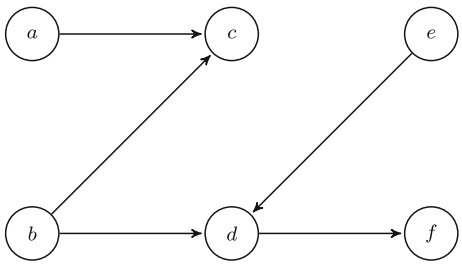


Fig. 7 A directed network that illustrates the difference between forward and backward simuation

We can also run the backward simulation algorithm on the network. This gives us the following equivalence classes $\{f\}$, $\{d, c\}$, $\{a, b, e\}$. Here there is a distinction between *f* and *c*. Recalling that backward simulation could also be considered as ordinary simulation equivalence but on a transformed network with the relation "receives e-mail from" instead, we realize that the two different simulation equivalences capture two distinct, equally important possible classifications of the nodes: in one case, how long is the e-mail chain inside the network before is reached a particular node, and in the other, how long is the e-mail chain after it has been sent from a specific node.

This illustrates one of the key points of the paper. *Different abstracted views or equivalence relations capture different features of a network*. Depending on the question that an analysts strives to answer, they will need to consider different equivalence relations. By subjecting the network under study to different equivalence relations, we can learn different things about it.

See Hanneman and Riddle (2005) for a well-written survey of the use of equivalence classes for identifying "positions" or "roles." This survey puts our work into perspective by defining structural equivalence, automorphic equivalence, and regular equivalence in terms of progressively generalized ways of partitioning the network into classes of interesting categories. However, similar to that of structural and automorphic equivalence, it is still the case that the more generic regular equivalence may be rare in a large population which, hence, calls for either approximate solutions or more generic definitions of equivalence—such as the definition presented in this paper.

4.5 Algorithm

A simple algorithm for computing the maximal simulation relation is shown in Fig. 8. For a directed graph $G = (V, E)$, the algorithm computes for each vertex $v \in V$ the simulation set $sim(v)$ that consists of the set of vertices that simulate *v*. Two states *u* and *v* are connected with an edge from *u* to *v* if the pair $(u, v) \in E$. For vertex *v*, we use the notation $pre(v)$ to denote the set of predecessors of

SIMULATION-RELATIONS

Input: a directed graph $G = (V, E)$

Output: for each vertex $v \in V$**, the simulator set** $sim(v)$

1 **for** all $v \in V$ **do** $sim(v) = V$
2 **while** there are three vertices u, v, and w such that
3 $v \in post(u), w \in sim(u)$, and $post(w) \cap sim(v) = \emptyset$ **do**
4 $sim(u) := sim(u) - \{w\}$

Fig. 8 An algorithm for computing the maximal simulation relation

$v : \{u \mid (u, v) \in E\}$. The notation $post(v)$ is used to denote the set of successors of $v : \{u \mid (v, u) \in E\}$.

As input, the algorithm takes a directed graph $G = (V, E)$. For each state v the set $sim(v)$ contains states that are candidates for simulating v. If the graph is unlabeled and unweighted then initially all states are candidates for simulating each other. The algorithm operates as follows: if there is an edge $(u, v) \in E$ and a state $w \in sim(u)$ but no state $w' \in sim(v)$ such that there exists an edge $(w, w') \in E$ then w cannot simulate u and is therefore removed from $sim(u)$. Two states v and u are simulation equivalent if $v \in sim(u)$ and $u \in sim(v)$.

The algorithm in Fig. 8 runs in time $O(M^2N^3)$ for a directed graph with N vertices and M edges. If the network has labels on the edges, complexity is increased by a factor as large as the size of the number of different labels.If the network has integers as weights (and we assume that comparisons can be done in constant time) the time complexity remains the same (Högberg et al. 2007).

There are several other algorithms for computing simulation relations that are faster than the algorithm in Fig. 8. We refer to for example Henzinger et al. (1995); Tan and Cleaveland (2001); Bustan and Grumberg (2000); Gentilini et al. (2002) and Francesco and Francesco (2007) for descriptions of more efficient, but also more complicated, implementations.

While simulation relations are computationally harder to compute than some other equivalences such as regular equivalence and structural equivalence, the complexity is still polynomial. This means that simulation relation can be computed in reasonable time for very large networks. A detailed exposition of experimental results for computing simulation relations are presented by Gentilini et al. (2003). If the size of the network is the number of vertices and the number of edges, then computing simulation on a networks of size 8,000 takes seconds and computing simulation on a network of size 350,000 takes minutes[1].

5 Experimental results 1: communication in a research department

Social networks play fundamental roles as mediums for spreading information, ideas and influence among their members. In the following examples, we test our ability to find social positions in networks where each tie represents the possibility to communicate. This example illustrates simulation equivalence, simulation preorder and how an abstraction of the original network can be created.

[1] The tests were executed on a Pentium III, 400 MHz PC, 256 MB RAM, OS Linux Red Hat 6.2.

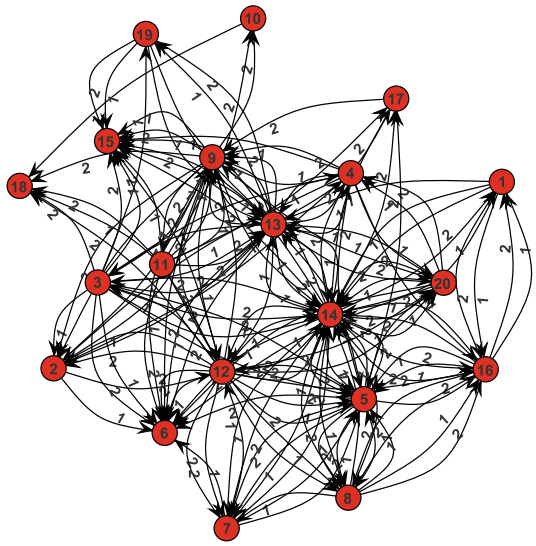


Fig. 9 The original network of communication

Simulation equivalence is compared with regular equivalence to illustrate the difference between the two equivalence relations.

5.1 Network of communication

In the experiments reported on here, we used a network obtained from the communication pattern within a department consisting of 20 researchers, see Fig. 9. The network is weighted and directed. A directed tie between two actors a and b represents the fact that actor a communicates with actor b.

In this experiment we assign a tie between two actors the weight one if their regular communication is done using e-mail and the weight two if the communication is done using phone. This means that communicating via phone is worth twice as much as communicating using e-mail, indicating that real-time talking signifies a closer social relation than that of virtual relationships.

Computing simulation equivalence on the the network we obtain a smaller network with five different equivalence classes, depicted in Fig. 10. The network in Fig. 10 contain *clusters of people* that are, to some extent, similar. If we compute regular equivalence on the same network according to Definition 7 we obtain 18 different equivalence classes divided into 17 classes containing one actor and one class containing 3 actors.

Using the preorder that was obtained when computing the simulation relation, we notice that one of the equivalence classes simulates all the other equivalence classes according to Table 1. Theoretically, this indicates that we can replace the other equivalence classes with this dominating equivalence class and still preserve all

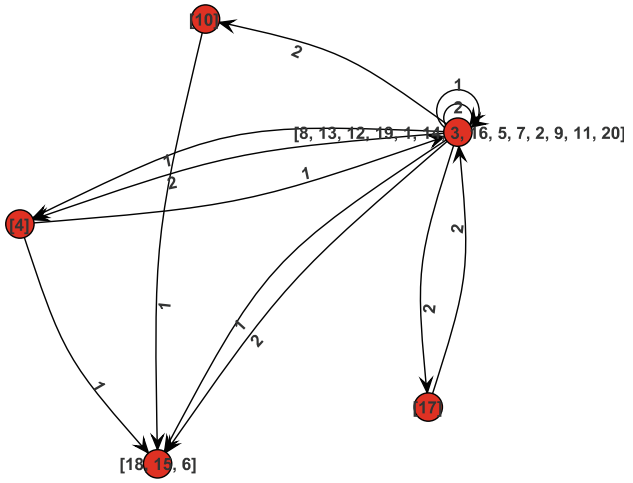


Fig. 10 Simulation equivalence computed on the network of communication

Table 1 This table summarizes all information about the simulation preorder

Actor	Simulated by actors
10	20, 17, 4, 16, 7, 12, 9, 10, 2, 14, 3, 11, 19, 1,5, 8, 13
4	20, 17, 4, 16, 7, 12, 9, 2, 14, 3, 11, 19, 1, 5, 8, 13
17	20, 17, 16, 7, 12, 9, 2, 14, 3, 11, 19, 1, 5, 8, 13
6, 15, 18	20, 15, 4, 16, 7, 12, 9, 10, 2, 14, 18, 3, 11, 19, 1, 5, 8, 13, 6
1, 2, 3, 5, 7, 8, 9, 11, 12, 13, 14, 16, 19, 20	1, 2, 3, 5, 7, 8, 9, 11, 12, 13, 14, 16, 19, 20

Each row shows an actor (to the left) and the set of actors that can simulate that actor (to the right)

communication that was present in the original network. Hence, it should be noted that the abstraction that can be obtained using the dominating equivalence class must be thought of in terms of *clusters of equivalence classes* rather than the clusters of individuals that are depicted in the graph in Fig. 10. Of course, this total reduction is due to the investigated social scenario (a research department) where all departmental members are, in some sense, comparable with respect to their duties. That is, since all members in the investigated organization are of the same type the graph collapses into one single node. In a more heterogeneous organization, the number of equivalence classes would of course correspond to the number of different types of personnel in the organization, e.g., a scientist cannot be assumed to be able to simulate an officer and vice versa since the jobs are so different.

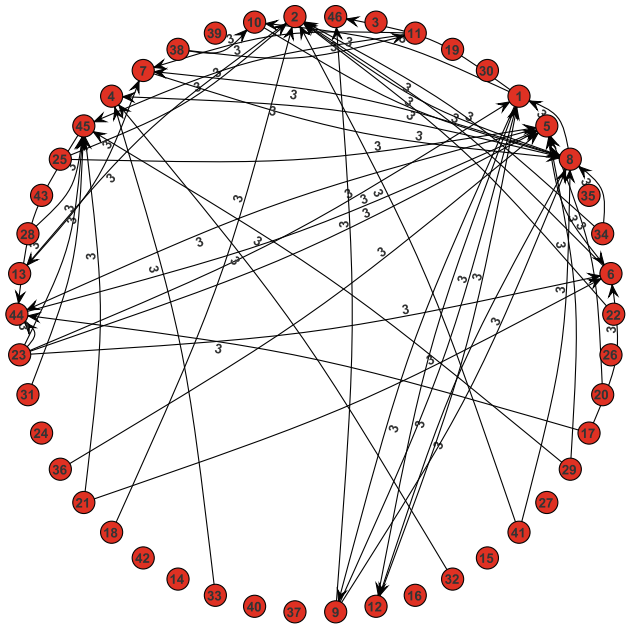


Fig. 11 The original network from the consulting company, using only the ties labeled with weight 3

6 Second example: relations in a consulting company

This example illustrates the use of simulation equivalence relations and the differences when the direction of ties are changed. The data set was used by Cross and Parker (2004). The data comes from a consulting company with 46 employees. The ties are differentiated on a scale from 0 to 5 in terms of frequency of information or advice requests. The weights on the ties have the following meaning:

(0) I do not know this person
(1) Never
(2) Seldom
(3) Sometimes
(4) Often
(5) Very often

In this experiment we have used a subset of the original network: all actors and the ties labeled with weight 3. The original network with ties labeled with the weight 3 is shown in Fig. 11. If there is a tie between an actor *a* and an actor *b* in the network, it means that *a* turns to *b* sometimes for information or advice on work-related issues.

Using simulation equivalence we obtain a network consisting of four different equivalence classes as shown in Fig. 12. The network has four different social positions, each represented by an equivalence class. However, if we reverse the directions of the ties and use simulation equivalence we obtain completely different equivalence classes. The resulting network using this backward simulation

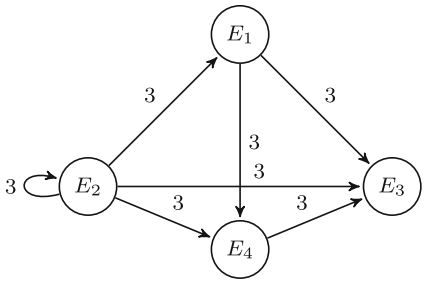


Fig. 12 Forward simulation equivalence computed on the network in Fig. 11

$$E_1 = \{45\}$$

$$E_2 = \{46, 7, 12, 13, 2, 4, 6, 8, 5, 44, 1, 9, 10\}$$

$$E_3 = \{31, 19, 41, 26, 25, 35, 38, 21, 29, 37, 30, 17, 39, 3, 15, 40, 20, 36, 14, 28, 32, 22, 27, 24, 33, 18, 42, 43, 16, 23, 34\}$$

$$E_4 = \{11\}$$

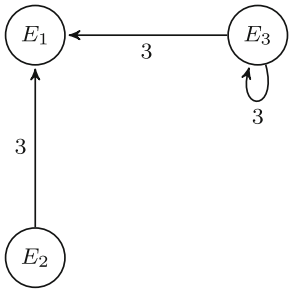


Fig. 13 Backward simulation equivalence computed on the network in Fig. 11

$$E_1 = \{46, 19, 26, 35, 37, 30, 39, 4, 3, 15, 40, 14, 27, 24, 42, 43, 16\}$$

$$E_2 = \{32, 33\}$$

$$E_3 = \{31, 41, 25, 7, 12, 38, 21, 13, 29, 45, 17, 2, 6, 20, 11, 8, 36, 28, 5, 22, 44, 18, 1, 9, 10, 23, 34\}$$

contains three different equivalence classes and is shown in Fig. 13.

In this example, the networks obtained using backward (reversed direction of ties) and forward simulation equivalence are completely different. In the case of forward simulation equivalence (Fig. 12) we have more equivalence classes and ties than if we use backward simulation equivalence (Fig. 13). The social positions differ depending on the direction of the ties. This is natural since in the forward case the social positions are dependent on whom the actors turn to for information while reversing the ties gives social positions that are dependent of the actors that are actually turned to. In this example, the networks obtained using backward (reversed direction of ties) and forward simulation equivalence are different. In the case of forward simulation equivalence (Fig. 12) we have more equivalence classes and ties than if we use backward simulation equivalence (Fig. 13). The equivalence classes (or social positions) differ when using the different simulation relations since the actors are grouped differently.

7 Application: military intelligence by graph reduction

Our primary application of interest is SNA within the military intelligence domain. Here, one wishes to consider and model a network of interesting people, e.g., a terrorist cell, that are connected to each other in various ways. In this paper we present results that are applicable to networks that are both directed and weighted, i.e., networks that are capable of modeling relations that are one-way and have an importance measure. That is, in these networks it is common that one person is related to another person but not the other way around, and the relations have an associated value so that the strength of relations can be valued vis-á-vis each other. In the intelligence domain, such networks typically arise as a result of homeland security and force protection applications. Here, the intelligence analyst is faced with the problem of modeling and understanding complex organizations to obtain an understanding of organizational structure, key roles, etc. To identify these key roles is a difficult task given a non-structured and concealed organization, but yet needed in order to, e.g.,

successfully perform information operations (Armistead 2004). Such measures are to be seen as natural steps towards preventing terrorist actions and require one to obtain an understanding of an organization that is actively trying to hide its structure and is unwilling to provide information regarding its operations and other doings. These organizations typically utilize various kinds of open communication networks to transmit their messages, meaning that intelligence personnel are likely to be in possession of graph data depicting the communication links but not the actual information that has been transmitted. Hence, the communication graph can often be built using information obtained from routing facilities or phone companies but the actual communication content is more difficult to obtain and/or decipher. Typically, the eavesdropped communication is of one-way/directed type, e.g., e-mail, mobile phone messaging, etc. Also, the actual kind of communication, its duration and its rate of recurrence give immediate information about the importance of the communication that can be used for edge weighting.

Military intelligence is largely an unknown business: the nature of the work and the resulting intelligence products make it vital to keep practices, methods and techniques secret. There is a dysfunctional and somewhat mythological side to this secrecy phenomenon as well, namely to protect the organization and its activities, and to preserve the assumed significance of trivial information and assessments (Agrell and Treverton 2009). As a consequence, it would be contradictory to think that intelligence analysis would ever "open up" to adopt overt, standardized, systematically employed, and verifiable methods. Nevertheless, on a generic level it is still apparent that a shift is currently taking place. Traditional intelligence work has been closely related to the so-called intelligence cycle where one plans, gathers documents, analyzes these documents and delivers a report. This iterative process is ill-suited to the modern information age and therefore new computerized methods making use of continuous updating, multiple sources and automation changes the very foundations of military intelligence work and turns the traditional way that analysts' work into a target-centric intelligence loop where several sources contribute in parallel to a continuously updated situation picture (Clark 2004). For example, the combination of manual social network analysis, live data from a mobile phone communication network and field observations can yield new insight and better intelligence products. One consequence of this shift is that the analyst is faced with networks containing large numbers of vertices and edges that need to be analyzed quickly and continuously. Hence, efficient graph reduction techniques and tools for graph mining are foreseen to be vital ingredients in tomorrow's computer support for intelligence analysts. Such data-analysis techniques are by no means a complete solution and do not replace the intuition and continual hypothesizing that are irreplaceable parts of the analytic process. However, data mining and related techniques are useful for early analysis and sorting tasks that would be impossible for human analysts, and also makes it possible to find patterns in data that humans could never detect without assistance (DeRosa 2004; Pollard 2009). In the end, therefore, we think of graph abstraction as a supporting tool—a tool out of many that can be very powerful when integrated in a larger "intelligence toolbox" (Brynielsson et al. 2009).

Given a large social network depicting a dark organization of some kind, the intelligence analyst can gain insight by finding graph patterns in many ways. The graph reduction techniques that we present in this paper help the analyst to discover important patterns within graph data that, in turn, give insights regarding important intelligence aspects regarding social roles and positions. Methodologically, the techniques that we present indicate three related types of graph reductions that can be used for three abstraction purposes: identification of equivalent actors, complexity reduction, and organizational structure visualization. First, any existing method used to detecting social roles and positions (such as regular equivalence or structural equivalence) can be used to group actors into clusters based on their similarity in the graph. For each actor in the graph, the similarity measure is obtained by looking at the graph properties which is hopefully a good approximation of the actor's actual skill or rank. That is, the underlying assumption is that an actor's communicative behavior reveals whether he/she is, e.g., a formal or informal leader at some level. This is an important part of intelligence analysis when studying, e.g., terrorist organizations: to cluster people depending on their formal or informal status. Second, *simulation equivalence* is a less strict similarity measure that says that two persons are simulation equivalent if they simulate each other. Since simulation equivalence is a less restrictive relation than the other suggested equivalence relations, different groups of actors will be formed and different social roles and positions can be detected. Since simulation equivalence can be defined both in terms of successors (forward simulation equivalence) or in terms of predecessors (backward simulation equivalence) a number of social roles and positions can be detected and depending on the question that an analysts strives to answer, the different equivalences may need to be considered. Finally, the *simulation relation* can be used to strip down each cluster into solely the most important actor within the cluster. Hence, simulation renders a graph that can no longer be thought of in terms of representing the original graph but rather as a means of illustrating how the clusters are related to each other, e.g., identifying leaders as opposed to subordinates and so forth.

8 Discussion and future work

In this paper we present a relation from computer science called *simulation relation* that can be used to distinguish between different social positions and roles in a social network. We argue that it is necessary to consider several different equivalence relations on a social network. Different equivalence relations will provide different perspectives on the network. If it is possible to define roles formally based on relations, it is also possible to detect emergent, unnamed roles and positions in social networks.

A simulation relation computed on the network N is an ordering (a preorder) of the nodes of N. The ordering is such that if an actor a simulates an actor b then actor a has at least the same relations as b to other actors (or actors that simulate these). Actors in the network that simulate each other are considered to be simulation equivalent and each such equivalence class represents a social position. The simulation relation can be computed based on successors or based on predecessors. When the relation is based on predecessors we denote it as backward simulation.

We use simulation equivalence to describe social positions within a social network since simulation equivalence is a less restrictive equivalence relation than regular equivalence, automorphic equivalence and structural equivalence.

We use the simulation preorder to create an abstraction of a given social network. The abstraction contains every trace that is found in the original network but the size may be significantly smaller than that of the original network. The abstraction can be used to perform a worst case scenario analysis of the network.

The social positions and the ordering that are obtained using simulation equivalence are particularly interesting when looking at social networks describing different competencies since we can use the positions to get an understanding of how competencies of the groups are composed.

We see many possibilities for future work based on simulation relations and ideas presented in this paper. First of all, it would be interesting to investigate if it is possible to define approximate simulation equivalence (as in the case of regular equivalence). Secondly, it would be interesting to conduct experiments on other, publically-available, data sources. In order to validate the usefulness of the relation for military intelligence analysis, it is necessary to implement the functionality into an SNA tool (such as the one used by (Ferrara et al. 2008) and conduct user experiments. Finally, it would be interesting to use the ideas in this paper to consider uncertain data. In this case, we might not know for sure that there is an edge between two actors; instead we only have a probability that there exists a link.

Another direction for future work is to conduct experiments with simulation relations on large social networks. Since there exist algorithms that runs in polynomial time and space, it should be possible to do experiments on large networks. One great challenge is to find a suitable meaningful data set (directed, weighted) to run the experiments on.

The authors will be happy to make the prototype source code available for further non-commercial experimental purposes. Please contact the authors if you are interested.

Acknowledgments This work was supported by the FOI research project "Tools for information management and analysis," which is funded by the R&D program of the Swedish Armed Forces, and by the ACTRESS project, which is funded by the Swedish Governmental Agency for Innovation Systems. We thank Christian Mårtenson and Johanna Högberg for discussions and contributions to this work.

References

Abdallah S (2011) Generalizing unweighted network measures to capture the focus in interactions. Soc Netw Anal Min. doi: 10.1007/s13278-011-0018-8

Agrell W, Treverton GF (2009) The science of intelligence: reflections on a field that never was. In: Treverton GF, Agrell W (eds) National intelligence systems: current research and future prospects. Cambridge University Press, New York, chap 11, pp 265–280

Armistead L (ed) (2004) Information operations: warfare and the hard reality of soft power. Issues in Twenty-First Century Warfare, Brassey's, Inc., Washington

Brynielsson J, Horndahl A, Kaati L, Mårtenson C, Svenson P (2009) Development of computerized support tools for intelligence work. In: 14th ICCRTS, Washington

Buchholz P (2008) Bisimulation relations for weighted automata. Theor Comput Sci 393(1–3):109–123

Bustan D, Grumberg O (2000) Simulation based minimization. In: Conference on automated deduction, pp 255–270

Carrington PJ, Scott J, Wasserman S (eds)(2005) Models and methods in social network analysis. Cambridge University Press, New York

Clark RM (2004) Intelligence analysis: a target-centric approach. CQ Press, Washington

Cross R, Parker A (2004) The hidden power of social networks: understanding how work really gets done in organizations. Harvard Business School Press, Boston

DeRosa M (2004) Data mining and data analysis for counterterrorism. CSIS Press, Washington

Everett MG, Borgatti SP (1994) Regular equivalence: general theory. J Math Soc 19(1):29–52

Fazeen M, Dantu R, Guturu P (2011) Identification of leaders, lurkers, associates and spammers in a social network: context-dependent and context-independent approaches. Soc Netw Anal Min. doi: 10.1007/s13278-011-0017-9

Ferrara L, Mårtenson C, Svenson P, Svensson P, Hidalgo J, Molano A, Madsen A (2008) Integrating data sources and network analysis tools to support the fight against organized crime. Lecture Notes in Computer Science, vol 5075. Springer, Berlin, pp 171–182

Francesco R, Francesco T (2007) A new efficient simulation equivalence algorithm. In: LICS '07: Proceedings of the 22nd

annual IEEE symposium on logic in computer science, pp 171–180
Freeman LC, White DR, Romney AK (eds) (1992) Research methods in social network analysis. Transaction Publishers, New Brunswick
Gentilini R, Piazza C, Policriti A (2002) Simulation as coarsest partition problem. In: Tools and algorithms for construction and analysis of systems, pp 415–430
Gentilini R, Piazza C, Policriti A (2003) From bisimulation to simulation: coarsest partition problems. J Autom Reason 31:73–103
Hanneman RA, Riddle M (2005) Introduction to social network methods. University of California, Riverside, chap 12
Henzinger MR, Henzinger TA, Kopke PW (1995) Computing simulations on finite and infinite graphs. In: Proceedings of the 36th annual symposium on foundations of computer science. IEEE Computer Society Press, pp 453–462
Högberg J, Maletti A, May J (2007) Bisimulation minimisation for weighted tree automata. In: Proceedings of 11th international conference developments in language theory. Lecture Notes in Computer Science, vol 4588, Springer, Berlin, pp 229–241
Hopcroft J (1971) An *nlogn* algorithm for minimizing states in a finite automaton. In: Kohavi Z (ed) Theory of machines and computations, Academic Press, New York
Kolaczyk ED (2009) Statistical analysis of network data: methods and models. Springer Series in Statistics. Springer, New York
Liggins ME, Hall DL, Llinas J (eds) (2009) Handbook of multisensor data fusion: theory and practice, 2nd edn. Electrical Engineering and Applied Signal Processing Series. CRC Press, Boca Raton
Lynch N, Vaandrager F (1995) Forward and backward simulations: I. Untimed systems. Inf Comput 121(2):214–233
Maletti A (2009) A backward and a forward simulation for weighted tree automata. In: Proceedings of 3rd international conference algebraic informatics
Marx M, Masuch M (2003) Regular equivalence and dynamic logic. Soc Netw 25:51–65
Newman M, Barabasi A, Watts DJ (eds) (2006) The structure and dynamics of networks. Princeton University Press, Princeton
Opsahl T, Panzarasa P (2009) Clustering in weighted networks. Soc Netw 31(2):155–163
Paige R, Tarjan R (1987) Three partition refinement algorithms. SIAM J Comput 16(6):973–989
Pollard NA (2009) On counterterrorism and intelligence. In: Treverton GF, Agrell W (eds) National intelligence systems: current research and future prospects. Cambridge University Press, New York, chap 6, pp 117–146
Scott J (2000) Social network analysis: a Handbook, 2nd edn. Sage Publications, London
Tan L, Cleaveland R (2001) Simulation revisited. Lecture Notes in Computer Science, vol 2031, pp 480–495
Wasserman S, Faust K (1994) Social network analysis: methods and applications. Cambridge University Press, Cambridge
Wolfe AW (2011) Anthropologist view of social network analysis and data mining. Soc Netw Anal Min 1(1):3–19