1

Reasoning about Complex Networks:
A Logic Programming Approach

PAULO SHAKARIAN [1]   GERARDO I. SIMARI [2]   DEVON CALLAHAN [1]

[1] *Network Science Center and Dept. of Electrical Engineering and Computer Science*
U.S. Military Academy, West Point, NY 10996, USA
[2] *Department of Computer Science, University of Oxford*
Wolfson Building, Parks Road, Oxford OX1 3QD, United Kingdom
(*e-mail:* `paulo@shakarian.net, gerardo.simari@cs.ox.ac.uk, devon.callahan@usma.edu`)

Abstract

Reasoning about complex networks has in recent years become an important topic of study due to its many applications: the adoption of commercial products, spread of disease, the diffusion of an idea, etc. In this paper, we present the MANCaLog language, a formalism based on logic programming that satisfies a set of desiderata proposed in previous work as recommendations for the development of approaches to reasoning in complex networks. To the best of our knowledge, this is the first formalism that satisfies all such criteria. We first focus on algorithms for finding minimal models (on which multi-attribute analysis can be done), and then on how this formalism can be applied in certain real world scenarios. Towards this end, we study the problem of deciding group membership in social networks: given a social network and a set of groups where group membership of only some of the individuals in the network is known, we wish to determine a degree of membership for the remaining group-individual pairs. We develop a prototype implementation that we use to obtain experimental results on two real world datasets, including a current social network of criminal gangs in a major U.S. city. We then show how the assignment of degree of membership to nodes in this case allows for a better understanding of the criminal gang problem when combined with other social network mining techniques—including detection of sub-groups and identification of core group members—which would not be possible without further identification of additional group members.

KEYWORDS: Knowledge Representation, Reasoning under Uncertainty, Complex Networks, Social Networks

1 Introduction and Related Work

An epidemic working through a population, cascading electrical power failures, product adoption, and the spread of a mutant gene are all examples of diffusion processes that can happen in complex networks. These network processes have been studied in a variety of disciplines, including computer science (Kempe et al. 2003), biology (Lieberman et al. 2005), sociology (Granovetter 1978), economics (Schelling 1978), and physics (Sood et al. 2008). Much existing work in this area is based on pre-existing models in sociology and economics—in particular the work of (Granovetter 1978; Schelling 1978). However, recent examinations of social networks—both analysis of large data sets and observational studies—have indicated that there may be additional factors to consider that are not

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **2013**	2. REPORT TYPE	3. DATES COVERED **00-00-2013 to 00-00-2013**
4. TITLE AND SUBTITLE **Reasoning about Complex Networks: A Logic Programming Approach**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Network Science Center and Dept. of Electrical,Engineering and Computer Science, U.S. Military Academy,,West Point,NY,10996**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT **Same as Report (SAR)**	18. NUMBER OF PAGES **14**	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**			

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

taken into account by these models. These include the attributes of nodes and edges, competing diffusion processes, and time. In this paper, we propose MANCaLog (*Multi-Attribute Networks and Cascades*), a logical language for modeling multi-attribute processes in complex networks that can richly express how individuals in the network adopt or fail to adopt certain behaviors, and how these behaviors diffuse through the network. MANCaLog is based on a set of design criteria recently proposed in (Shakarian et al. 2013), and it is to the best of our knowledge the first logical language for modeling diffusion in complex networks that meets these criteria. We also introduce fixed-point based algorithms for computing the result of a diffusion process. Note that these algorithms are proven not only to be correct, but also to run in polynomial time. Hence, our approach can not only better express many aspects of multi-attribute processes in complex networks, but it can do so in a reasonable amount of time. Finally, we investigate applications by considering the problem of deciding group membership in social networks: given a social network and a set of groups where membership of only some of the individuals is known, we wish to determine a degree of membership for the remaining group-individual pairs. We also develop a prototype implementation that we use to obtain experimental results on two real world datasets, including a current social network of criminal gangs in a major U.S. city.

1.1 Design Criteria

In recent work (Shakarian et al. 2013), we proposed a set of seven design criteria that we believe a framework for reasoning about multi-attribute processes in complex networks should satisfy. As a quick overview, these criteria are: (i) Multiply labeled and weighted nodes and edges: Many existing frameworks for studying diffusion in complex networks assume that there is only one type of vertex that may become "active" or may "mutate" and only one possible relationship between nodes; however, in reality nodes and edges often have different properties. For instance, labels on edges can be used to differentiate between strong and weak ties (edge types); (ii) Explicit representation of time: Most work in the literature either assumes static models or makes several simplifying assumptions such as a model of time solely based on temporal decay of influence; we seek a richer model of temporal relationships between conditions in the network structure, the current state of the cascades in process, and how influence propagates; (iii) Non-Markovian temporal relationships: Temporal dependencies should be able to span multiple units of time; hence, the "memoryless" mode of a standard Markov process is insufficient. We strive to create a framework where dependencies can be from other earlier time steps; (iv) Representation of uncertainty: In practice, it is not always possible to judge the attributes of all individuals in a network, and thus an element of uncertainty must be included. In connection with point (vii) below, this should not be at the expense of tractability; (v) Competing processes: Real-world situations often present competing network processes, where the success of one hinges on the failure of the other; (vi) Non-Monotonic Processes: Though in much existing work on diffusion processes in complex networks the number of nodes attaining a certain property at each time step can only increase, if we allow for competing cascades in the same model we cannot have such a strong restriction; and (vii) Tractability: The social networks of interest in today's data mining problems often have millions of nodes, and it is reasonable to expect that soon billion-node networks

Criterion	MANCaLog	IC/LT	SNOP	CD	EGT/VM
1. Labels	Yes	No	Yes	Yes	No
2. Explicit Representation of Time	Yes	No	Yes	No	Yes
3. Non-Markovian Time	Yes	No	No	No	No
4. Uncertainty	Yes	Yes	Yes	Yes	Yes
5. Competing Processes	Yes	No	No	Yes	Yes
6. Non-monotonic Processes	Yes	No	No	Yes	Yes
7. Tractablity	PTIME	#P-hard	PTIME	PTIME	NP-hard

Fig. 1. A comparison of models.

will be commonplace. Any framework for dealing with these problems must be tractable and offer areas for practical improvement for further scalability.

1.2 Related Work

The above criteria can be summarized as the desire to design the most expressive language for network cascades possible while still allowing computation of the outcome of a diffusion process to be completed in a tractable amount of time. As a comparison, let us briefly describe some relevant related work. Perhaps the best known general model for representing diffusion in complex networks is the independent cascade/linear threshold (IC/LT) model of (Kempe et al. 2003). However, although this framework was shown to be capable of expressing a wide variety of sociological models, it assumes the Markov property and does not allow for the representation of multiple attributes on vertices and edges. A more recent framework, social network optimization problems (SNOPs) (Shakarian et al. 2010) uses logic programming to allow for the representation of attributes, but this framework does not allow for competing processes or non-monotonic cascades. A related logic programming framework, competitive diffusion (CD) (Broecheler et al. 2010) allows for competitive diffusion and non-monotonic processes but does not explicitly represent time and also makes Markovian assumptions. Further, we also note that the semantics of CD yields a "most probable interpretation" that is not a unique solution. Hence, a given model in that framework can lead to multiple and possibly contradictory, outcomes to a cascade (this problem is avoided in MANCaLog). Another popular class of models is Evolutionary Graph Theory (EGT) (Lieberman et al. 2005), which is highly related to the voter model (VM) (Sood et al. 2008). Although this framework allows for competing processes and non-monotonic diffusion, it also makes Markovian assumptions while not explicitly representing time. Further, determining the outcome of a cascade in those models is NP-hard, while determining the outcome in MANCaLog can be accomplished in polynomial time. Table 1 lists how these models compare to MANCaLog when considering our design criteria.

The rest of this paper is organized as follows: Section 2 presents the MANCaLog framework; Section 3 discusses consistency, entailment, and fixpoint computation of minimal models; Section 4 discusses applications in social networks and experimental results, and Section 5 includes conclusions and future work.
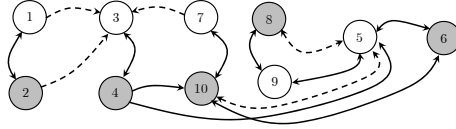
Fig. 2. Simple online social network G_{soc}. Solid edges are labeled with *strTie* and dashed edges with *wkTie*. White nodes are labeled with *male* and gray nodes with *fem*. Arrows represent the direction of the edge; double-headed edges represent two edges with the same label.

2 The MANCaLog Language: Syntax and Semantics

In this work we assume that individuals (persons, agents, etc.) are arranged in a directed graph (or network) $G = (V, E)$, where the set of nodes corresponds to the individuals, and the edges model the relationships between them. We also assume a set of labels $\mathcal{L}$, which is partitioned into two sets: *fluent* labels $\mathcal{L}_f$ (labels that can change over time) and *non-fluent* labels $\mathcal{L}_{nf}$ (labels that do not); labels can be applied to both the nodes and edges of the network. We will use the notation $\mathcal{G} = V \cup E$ to be the set of all *components* (nodes and edges) in the network. Thus, $c \in \mathcal{G}$ could be either a node or an edge.

Example 2.1
We will use the sample online social network G_{soc} shown in Figure 2 as the running example; $\mathcal{G}_{soc}$ is used to denote the set of components of G_{soc}. Here we have $\mathcal{L}_{nf} = \{male, fem, strTie, wkTie\}$ representing male, female, strong ties and weak ties, respectively. Additionally, we have $\mathcal{L}_f = \{visPgA, visPgB\}$ representing visiting webpage A and visiting webpage B, respectively. ∎

We now present a logical language where we use atoms, referring to labels and weights, to describe properties of the nodes and edges. Though labels themselves could be modeled as atoms instead of predicates (to model non-ground labelings that allow for greater expressibility), for simplicity of presentation we leave this to future work. The first piece of the syntax is the **network atom**.

Definition 2.1 (Network Atom)
Given label $L \in \mathcal{L}$ and real-valued interval $bnd \subseteq [0, 1]$ (referred to as a "weight interval"), a *network atom* is of the form $\langle L, bnd \rangle$. A network atom is *fluent* (resp., *non-fluent*) if $L \in \mathcal{L}_f$ (resp., $L \in \mathcal{L}_{nf}$). The set of possible network atoms is denoted with NA.

Network atoms describe properties of nodes and edges. The definition is intuitive: L represents a property of the vertex or edge, and associated with this property is some weight that may have associated uncertainty—hence represented as an interval bnd, which can be open or closed. An invalid bound is represented by $\emptyset$, which is equivalent to all other invalid bounds.

Definition 2.2 (World)
A *world* W is a set of network atoms such that for each $L \in \mathcal{L}$ there is no more than one network atom of the form $\langle L, bnd \rangle$ (where $bnd \neq \emptyset$) in W.

A **network formula** over NA is defined using conjunction, disjunction, and negation in the usual way. If a formula contains only non-fluent (resp., fluent) atoms, it is a non-fluent (resp., fluent) formula.

Definition 2.3 (*Satisfaction of Worlds*)
Given world W and network formula f, *satisfaction* of W by f is defined as follows:
− If $f = \langle L, [0,1] \rangle$ then $W \models f$.
− If $f = \langle L, \emptyset \rangle$ then $W \not\models f$.
− If $f = \langle L, bnd \rangle$, with $bnd \neq \emptyset$ and $bnd \neq [0,1]$, then $W \models f$ iff there exists $\langle L, bnd' \rangle \in W$ s.t. $bnd' \subseteq bnd$.
− If $f = \neg f'$ then $W \models f$ iff $W \not\models f'$.
− If $f = f_1 \wedge f_2$ then $W \models f$ iff $W \models f_1$ and $W \models f_2$.
− If $f = f_1 \vee f_2$ then $W \models f$ iff $W \models f_1$ or $W \models f_2$.

For some arbitrary label $L \in \mathcal{L}$, we will use the notation $\mathsf{Tr} = \langle L, [0,1] \rangle$ and $\mathsf{F} = \langle L, \emptyset \rangle$ to represent a tautology and contradiction, respectively. For ease of notation (and without loss of generality), we say that if there does not exist some bnd s.t. $\langle L, bnd \rangle \in W$, then this implies that $\langle L, [0,1] \rangle \in W$.

Example 2.2
Following from Example 2.1, the network atom $\langle female, [1,1] \rangle$ can be used to identify a node as a woman. World $W = \{ \langle fem, [1,1] \rangle, \langle male, [0,0] \rangle, \langle visPgA, [1,1] \rangle, \langle visPgB, [0,0] \rangle \}$ might be used to identify a woman who visits webpage A. Clearly, we have that $W \models \langle fem, [1,1] \rangle \wedge \neg \langle visPgA, [0.5, 0.9] \rangle \wedge \neg \langle visPgB, [0.1, 0.7] \rangle$. ∎

The idea is to use MANCaLog to describe how properties (specified by labels) of the nodes in the network change over time. We assume that there is some natural number t_{max} that specifies the total amount of time we are considering, and we use $\tau = \{ t \mid t \in [0, t_{max}] \}$ to denote the set of all time points. How well a certain property can be attributed to a node is based on a *weight* (to which the *bnd* bound in the network atom refers). As time progresses, a weight can either increase/decrease and/or become more/less certain. We now introduce the MANCaLog fact, which states that some network atom is true for a node or edge during certain times.

Definition 2.4 (*MANCaLog Fact*)
If $[t_1, t_2] \subseteq [0, t_{max}]$, $c \in \mathcal{G}$, and $a \in NA$, then $(a, c) : [t_1, t_2]$ is a MANCaLog **fact**. A fact is *fluent* (resp., *non-fluent*) if atom a is fluent (resp., non-fluent). All non-fluent facts must be of the form $(a, c) : [0, t_{max}]$. Let $\mathcal{F}$ be the set of all facts and $\mathcal{F}_{nf}, \mathcal{F}_f$ be the set of all non-fluent and fluent facts, respectively.

An example of a fact based on the running example is $F = (\langle male, [1,1] \rangle, 1) : [0, t_{max}]$. Next, we introduce integrity constraints (ICs).

Definition 2.5 (*Integrity constraint*)
Given fluent network atom a and conjunction of network atoms b, an *integrity constraint* is of the form $a \hookleftarrow b$.

Intuitively, integrity constraint $\langle L, bnd \rangle \hookleftarrow b$ means that if at a certain time point a component (vertex or edge) of the network has a set of properties specified by conjunction b, then at that same time the component's weight for label L must be in interval bnd. Following from the previous examples, the integrity constraint $\langle male, [0,0] \rangle \hookleftarrow \langle fem, [1,1] \rangle$ would require any node designated as a female to not be male.

We now turn to MANCaLog rules. The idea behind rules is simple: a node that meets some criteria is influenced by the set of its neighbors who possess certain properties.

The amount of influence exerted on a node by its neighbors is specified by an *influence function*, whose precise effects will be described later on when we discuss the semantics. As a result, a rule consists of four major parts: (i) an influence function, (ii) neighbor criteria, (iii) target criteria, and (iv) a target. Intuitively, (i) specifies how the neighbors influence the node in question, (ii) specifies which of the neighbors can influence the node, (iii) specifies the criteria that cause the node to be influenced, and (iv) is the property of the node that changes as a result of the influence.

We will discuss each of these parts in turn, and then define rules in terms of these elements. First, we define influence functions and neighbor criteria.

Definition 2.6 (Influence Function)
An *influence function* is a function $ifl : \mathbf{N} \times \mathbf{N} \to [0,1] \times [0,1]$ that satisfies the following two axioms:

1. *ifl* can be computed in constant ($O(1)$) time.

2. For $x' > x$ we have $ifl(x', y) \subseteq ifl(x, y)$.

We use *IFL* to denote the set of all influence functions.

Intuitively, an influence function takes the number of qualifying influencers (those that meet some requirement to be able to influence a certain individual, yet may or may not carry a contagion) and the number of eligible influencers (those that meet some requirement to be able to influence a certain individual *and* carry a contagion) and returns a bound on the new value for the weight of the property of the target node that changes. In practice, we expect the time complexity of such a function to be polynomial in terms of its arguments. However, as both arguments are naturals bounded by the maximum degree of a node in the network, this value will be much smaller than the size of the network—we thus treat it as a constant here.

Definition 2.7 (Neighbor Criterion)
If g_{edge}, g_{node} are non-fluent network formulas, h is a conjunction of network atoms, and *ifl* is an influence function, then $(g_{edge}, g_{node}, h)_{ifl}$ is a *neighbor criterion*.

Formulas g_{node} and h in a neighbor criterion specify the (non-fluent and fluent, respectively) criteria on a given neighbor, while formula g_{edge} specifies the non-fluent criteria on the directed edge from that neighbor to the node in question.

The next component is the "target criteria", which are the conditions that a node must satisfy in order to be influenced by its neighbors. Ideas such as "susceptibility" (Aral and Walker 2012) can be integrated into our framework via this component. We represent these criteria with a formula of non-fluent network atoms. The final component, the "target", is simply the label of the target node that is influenced by its neighbors. Hence, we now have all the pieces that comprise a rule.

Definition 2.8 (Rule)
Given fluent label L, natural number Δt, target criteria f and neighbor criteria $(g_{edge}, g_{node}, h)_{ifl}$, a *MANCaLog rule* is of the form: $r = L \xleftarrow{\Delta t} f, (g_{edge}, g_{node}, h)_{ifl}$. We will use the notation $head(r)$ to denote L.

Note that the target (also referred to as the head) of the rule is a single label; essentially, the body of the rule characterizes a set of nodes, and this label is the one that is modified

for each node in this set. More specifically, the rule states that when certain conditions for a node and its neighbors are met, the *bnd* bound for the network atom formed with label L on that node changes. Later, in the semantics, we introduce network interpretations, which map components (nodes and edges) of the network to worlds at a given point in time. The rule dictates how this mapping changes in Δt time steps.

Definition 2.9 (MANCaLog Program)
A *program* P is a set of rules, facts, and integrity constraints s.t. each non-fluent fact $F \in \mathcal{F}_{nf}$ appears no more than once in the program. Let **P** be the set of all programs.

Example 2.3
Following from the running example, suppose $sftTp$ and $ngTp$ are influence functions. Consider the following rules:

$$
\begin{aligned}
R_1 &= visPgA \overset{2}{\leftarrow} \langle fem, [1,1] \rangle, (\langle strTie, [0.9,1] \rangle, \mathsf{Tr}, \langle visPgA, [0.9,1.0] \rangle)_{sftTp} \\
R_2 &= visPgB \overset{1}{\leftarrow} \langle male, [1,1] \rangle, (\mathsf{Tr}, \mathsf{Tr}, \langle visPgB, [0.8,1.0] \rangle)_{sftTp} \\
R_3 &= visPgA \overset{3}{\leftarrow} \langle male, [1,1] \rangle, (\mathsf{Tr}, \langle fem, [1,1] \rangle, \neg \langle visPgA, [0.7,1.0] \rangle)_{ngTp}
\end{aligned}
$$

Rule R_1 says that a female node in the network visits page A with a weight specified by the $sftTp$ influence function if at a certain number of her strong ties (with weight of at least 0.9) visited the page two days ago. The rest of the rules can be read analogously. ∎

Semantics. We now introduce our first semantic structure: the *network interpretation*.

Definition 2.10 (Network Interpretation)
A *network interpretation* is a mapping of network components to sets of network atoms, $NI : \mathcal{G} \to 2^{NA}$. We will use **NI** to denote the set of all network interpretations.

Note that not all labels will necessarily apply to all nodes and edges in the network. For instance, certain labels may describe a relationship while others may only describe a property of an individual. If a given label L does not describe a certain component c of the network, then in a valid network interpretation NI, $\langle L, [0,1] \rangle \in NI(c)$. We define a MANCaLog interpretation (simply referred to as "interpretation") as follows.

Definition 2.11 (Interpretation)
A *MANCaLog interpretation* I is a mapping of natural numbers in the interval $[0, t_{max}]$ to network interpretations, i.e., $I : \mathbf{N} \to \mathbf{NI}$. Let $\mathcal{I}$ be the set of all possible interpretations.

We now need to define *satisfaction* of the basic elements by interpretations. First, we define what it means for an interpretation to satisfy a fact and a rule.

Definition 2.12 (Fact Satisfaction)
An interpretation I *satisfies* fact $(a, c) : [t_1, t_2]$, written $I \models (a, c) : [t_1, t_2]$, iff $\forall t \in [t_1, t_2]$, $I(t)(c) \models a$.

For non-fluent facts, we introduce the notion of strict satisfaction, which enforces the bound in the interpretation to be set to exactly what the fact dictates.

Definition 2.13 (Strict Fact Satisfaction)
Interpretation I *strictly satisfies* fact $(a, c) : [t_1, t_2]$ iff $\forall t \in [t_1, t_2]$, $a \in I(t)(c)$.

Next, we define what it means for an interpretation to satisfy an integrity constraint.

Definition 2.14 (IC Satisfaction)
An interpretation I *satisfies* integrity constraint $a \hookleftarrow b$ iff for all $t \in \tau$ and $c \in \mathcal{G}$, $I(t)(c) \models \neg b \lor a$.

Before we define rule satisfaction, we require two auxiliary definitions that are used to define the bound enforced on a label by a given rule, and the set of time points that are affected by a rule.

Definition 2.15 (Bound function)
For a given rule $r = L \xleftarrow{\Delta t} f, (g_{edge}, g_{node}, h)_{ifl}$, node v, and network interpretation NI, $Bound(r, v, NI) = ifl(|Qual(v, g_{edge}, g_{node}, h, NI)|, |Elig(v, g_{edge}, g_{node}, NI)|)$, where we have $Elig(v, g_{edge}, g_{node}, NI) = \{v' \in V \mid NI(v') \models g_{node} \land (v', v) \in E \land NI((v', v)) \models g_{edge}\}$ and $Qual(v, g_{edge}, g_{node}, h, NI) = \{v' \in Elig(v, g_{edge}, g_{node}, NI) \mid NI(v') \models h\}$.

Intuitively, the bound returned by the function depends on the influence function and the number of qualifying and eligible nodes that influence it.

Definition 2.16 (Target Time Set)
For interpretation I, node v, and rule $r = L \xleftarrow{\Delta t} f, (g_{edge}, g_{node}, h)_{ifl}$, the *target time set* of I, v, r is defined as: $TTS(I, v, r) = \{t \in [0, t_{max}] \mid I(t - \Delta t)(v) \models f\}$ We also extend this definition to a program P, for a given $c \in \mathcal{G}$ and $L \in \mathcal{L}$, as follows; $TTS(I, c, L, P) = \bigcup_{r \in P, head(r) = L} TTS(I, c, r) \cup \{t \in [t_1, t_2] \mid (\langle L, bnd \rangle, c) : [t_1, t_2] \in P\} \cup \{t \mid \langle L, bnd \rangle \hookleftarrow b \in P \land I(t)(c) \models b\}$

We can now define satisfaction of a rule by an interpretation.

Definition 2.17
An interpretation I *satisfies* a rule $r = L \xleftarrow{\Delta t} f, (g_{edge}, g_{node}, h)_{ifl}$ iff for all $v \in V$ and $t \in TTS(I, v, r)$ it holds that $I(t)(v) \models \langle L, Bound(r, v, I(t - \Delta t)) \rangle$.

We now define satisfaction of programs, and introduce *canonical interpretations*, in which time points that are not "targets" retain information from the last time step.

Definition 2.18 (Models and Canonical models)
For interpretation I and program P:

I is a *model* for P iff it satisfies all rules, integrity constraints, and fluent facts in that program, strictly satisfies all non-fluent facts in the program, and for all $L \in \mathcal{L}$, $c \in \mathcal{G}$ and $t \notin TTS(I, c, L, P)$, $\langle L, [0, 1] \rangle \in I(c)(t)$.

I is a *canonical model* for P iff it satisfies all rules, integrity constraints, and fluent facts in P, strictly satisfies all non-fluent facts in P, and for all $L \in \mathcal{L}$, $c \in \mathcal{G}$, and $t \notin TTS(I, c, L, P)$, $\langle L, [0, 1] \rangle \in I(c)(t)$ when $t = 0$ and $\langle L, bnd \rangle \in I(t)(c)$ where $\langle L, bnd \rangle \in I(t - 1)(c)$.

3 Consistency, Entailment, and Fixpoint Model Computation

In this section we discuss consistency and entailment in MANCaLog programs, and explore the use of minimal models towards computing answers to these problems.

Definition 3.1 (Consistency and entailment)
A MANCaLog program P is *(canonically) consistent* iff there exists a (canonical) model I of P. P (canonically) entails MANCaLog fact F iff for all (canonical) models I of P, it holds that $I \models F$.

Now we define an ordering over models and define the concept of minimal model. We then show that if we can find a minimal model then we can answer consistency, entailment, and tight entailment queries. We first define a pre-order over interpretations.

Definition 3.2 (Preorder over interpretations, equivalence, and partial ordering)
Given interpretations I, I' we say $I \sqsubseteq^{pre} I'$ iff for all t, v, L if there exists $\langle L, bnd \rangle \in I(t)(v)$ then there must exist $\langle L, bnd' \rangle \in I'(t)(v)$ s.t. $bnd' \subseteq bnd$.

I, I' are **equivalent** (written $I \sim I'$) iff for all $P \in \mathbf{P}$, $I \models P$ iff $I' \models P$.

Given classes of interpretations $[I], [I']$ that are equivalent w.r.t. $\sim$, we say that $[I]$ precedes $[I']$, written $[I] \sqsubseteq [I']$, iff $I \sqsubseteq^{pre} I'$.

Definition 3.3 (Minimal Model)
Given program P, the *minimal model* of P is a (canonical) interpretation I s.t. $I \models P$ and for all (canonical) interpretation I' s.t. $I' \models P$, we have that $I \sqsubseteq I'$.

We can think of a minimal model of a MANCaLog program as the outcome of a multi-attribute process in a complex network that allows us to answer any entailment query.

Fixpoint Model Computation. We now introduce a fixed-point operator that produces the non-canonical minimal model of a MANCaLog program in polynomial time; first, we introduce three preliminary definitions.

Definition 3.4
Given program P, interpretation I, $c \in \mathcal{G}$, $L \in \mathcal{L}$, and $t \in \tau$, we define functions:
- $FBnd(P, c, t, L) = \bigcap_{(\langle L, bnd \rangle, c):[t_1, t_2] \in P \ s.t. \ t \in [t_1, t_2]} bnd$
- $IBnd(P, c, t, L) = \bigcap_{\langle L, bnd \rangle \leftarrow a \in P \ s.t. \ I(t)(c) \models a} bnd$
- $RBnd(P, I, v, t, L) = \bigcap_{r \in P \ s.t. \ t \in TTS(I, v, L, P) \cap TTS(I, v, r)} Bound(r, v, I(t - \Delta t))$.

We can now introduce the operator.

Definition 3.5 (Γ Operator)
For a given MANCaLog program P, we define the operator $\Gamma_P : \mathcal{I} \to \mathcal{I}$ as follows: For a given I, for each $t \in \tau$, $c \in \mathcal{G}$, and $L \in \mathcal{L}$, add $\langle \mathcal{L}, bnd \rangle$ to $\Gamma_P(I)(t)(c)$ where bnd is defined as: $bnd = bnd_{prv} \cap FBnd(P, c, t, L) \cap IBnd(P, I, c, t, L) \cap RBnd(P, I, c, t, L)$, where $\langle L, bnd_{prv} \rangle \in I(t)(c)$.

It is easy to show that Γ can be computed in polynomial time. Next, we introduce notation for repeated applications of Γ.

Definition 3.6 (Iterated Applications of Γ)
Given natural number $i > 0$, interpretation I, and program P, we define $\Gamma_P^i(I)$, the multiple applications of Γ: $\Gamma_P^i(I) = \Gamma_P(I)$ if $i = 1$ and $\Gamma_P^i(I) = \Gamma_P(\Gamma_P^{i-1}(I))$ otherwise.

The iterated Γ operator converges after a polynomial number of applications:

Theorem 3.1
Given interpretation I and program P, there exists a natural number k s.t. $\Gamma_P^k(I) = \Gamma_P^{k+1}(I)$, and $k \in O(|P| \cdot d_*^{in} \cdot t_{max} \cdot |E|)$ where d_*^{in} is the maximum in-degree in the network.

In the following, we will use the notation Γ_P^* to denote the iterated application of Γ after a number of steps sufficient for convergence; Theorem 3.1 means that we can efficiently compute Γ_P^*. We also note that as a single application of Γ can be computed in polynomial time, this implies that we can find a minimal model of a MANCaLog program in polynomial time. We now prove the correctness of the operator. We do this first by proving a key lemma that, when combined with a claim showing that for consistent program P, Γ_P^* is a model of P, tells us that Γ_P^* is a minimal model for P. Following directly from this, we have that P is inconsistent iff $\Gamma_P^* = \top$.

Lemma 3.1
If $I \models P$ and $I' \sqsubseteq I$ then $\Gamma(I') \sqsubseteq I$.

Theorem 3.2
If program P is consistent then Γ_P^* is a minimal model for P.

These results, when taken together, prove that tight entailment and consistency problems for MANCaLog can be solved in polynomial time, which is precisely what we set out to accomplish as part of our desiderata described in Section 1.

4 An Application in Social Networks: Discussion and Experimental Results

An important problem with regard to social networks is to determine group membership of the nodes (individuals). In particular, we are interested in the problem where some of the individuals in the network have been identified as members of a particular group while the affiliation of the remainder is unknown. In our work with a major U.S. metropolitan police force, we have found this to be an important problem in combating gang violence. Since in most cases it is considered a criminal offense to simply be in a gang, many gang members deny any type of affiliation upon arrest. Hence, in order to better understand the dynamics of these criminal organizations, it becomes necessary to use the data at hand to try to identify those with unknown affiliation. One way in which this can be done is by using MANCaLog rules that assign a *degree of membership* for each group to each individual with an unknown affiliation; this degree is a number in the interval $[0, 1]$ that specifies the confidence that they are in that group.

To address this problem, we propose the following. Consider a social network of individuals (for the police, this network is created based on co-arrestee data). Each group is assigned a fluent label and, for this problem, only one time point is used. For each node i that is in a group g, we include the fact $(\langle g, [1, 1] \rangle, i) : [0, 0]$ and, for each $g' \neq g$, the fact $(\langle g', [0, 0] \rangle, i) : [0, 0]$. For all other nodes j we include the fact $(\langle g, [0, 1] \rangle, j) : [0, 0]$ for each group g. We used a simple algorithm (not included due to space constraints) which creates influence functions and rules that assign degrees of membership based on the number of adjacent nodes within a given group. Then, by using the Γ function, we can compute the degree of membership for nodes with an unknown affiliation.
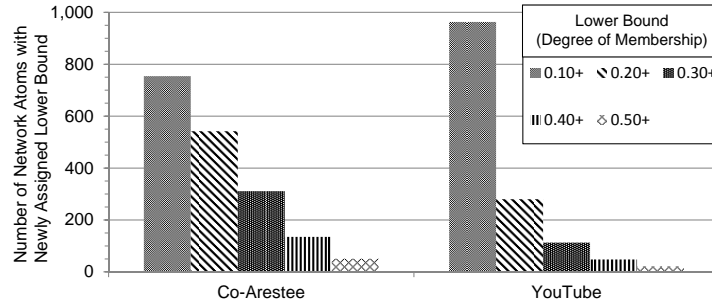
Fig. 3. Histogram illustrating number of network atoms assigned a lower bound of greater than zero after the convergence of Γ (omitting network atoms assigned an initial lower bound of 1.0).
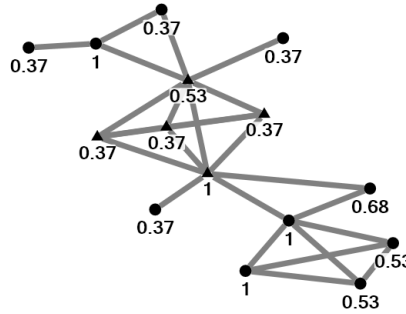

Fig. 4. Visualization of a subgroup of a faction. Lower bound of degree of membership is shown. Core members are denoted with a triangle.

Implementation and Experimental Results. We implemented the Γ operator and the computation of its fixed point in Python 2.7.3 in 700 lines of code that leveraged the NetworkX library[1]. Additionally, we implemented a rule-learning algorithm and supporting routines in an additional 300 lines of code. The experiments were run on a computer equipped with an Intel X5677 Xeon Processor operating at 3.46 GHz with a 12 MB Cache running Red Hat Enterprise Linux version 6.1 and equipped with 70 GB of physical memory.

We used two datasets: the previously described gang co-arrestee dataset provided by the police force of a major U.S. city, and a network derived from YouTube (based on channel subscriptions) (Yang and Leskovec 2012). The co-arrestee dataset consists of $2,333$ nodes and $3,676$ edges. The program used for this dataset consists of 58 rules. The YouTube dataset consists of $1,134,890$ nodes and $2,987,624$ edges, and we used a program with 47 rules. We note that the running time for the convergence of the Γ operator for the co-arrestee dataset was 38.41 seconds, while the running time for the much larger YouTube dataset was 63.6 hours; though this may be considered a long time, note that it is a one-time computation that allows us to answer many queries once the structure is obtained.

In Figure 3 we illustrate the number of nodes in the network whose lower bound on

---

[1] http://networkx.lanl.gov/

degree of membership for any of the groups increased after computing the convergence of Γ. Note that in our target application, we were able to assign a non-zero degree of membership to several hundred nodes. With rare exceptions, for the co-arrestee network, nodes were assigned a degree of membership to only one group (gang faction).

In order to get an understanding of the utility of assigning degree of membership, we consider the results of the convergence of the Γ operator used as input for some common social network analysis techniques that are likely to aid in police operations. We examine the sub-graph induced by individuals who had a degree of membership greater than or equal to 0.3 (a value chosen subjectively given the setup) for a certain gang faction. We then used the Louvain algorithm (Blondel et al. 2008) (modularity-maximizing) to identify sub-groups of that faction. The identification of sub-groups of such factions is useful to police to better understand the structure and dynamics of these organizations in order to improve law enforcement operations. The sub-graph induced by one such sub-group is shown in Figure 4. Note that the majority of the members in this sub-group have a degree of membership in the faction less than 1, which means that they were assigned by the Γ operator. This tells us that the sub-group might have been overlooked if degrees of membership were not being computed. Also, many of the individuals designated as "core members" (shown with a triangle in the figure) based on shell decomposition (Seidman 1983) were also individuals whose degree of membership was determined by Γ. Based on the work of (Kitsak et al. 2010), core members are thought to be key spreaders of information and thus also of interest for policing operations, particularly with regard to gathering intelligence on the sub-group in question.

5 Conclusions and Future Work

In this paper, we presented the MANCaLog language for describing multi-attribute networks and cascades. We started by recalling seven criteria in the form of desiderata for such a formalism, and showed that MANCaLog meets all of them; to the best of our knowledge, this has not been accomplished by any previous model in the literature. We also implemented this language and applied it to the degree of membership problem in social networks and showed how the results can aid in real-world law enforcement operations. We also note that MANCaLog is the first language of its kind to consider network structure in the semantics, potentially opening the door for algorithms that leverage features of network topology to more efficient query answering algorithms.

Currently, we are looking at other applications of MANCaLog as well as methods to learn rules that describe diffusion processes in social networks. In the near future, we shall also explore various types of queries that have been studied in the literature, such as finding nodes of maximum influence, identifying nodes that cause a cascade to spread more quickly, and identifying nodes that can be influenced in order to halt a cascade.

References

ARAL, S. AND WALKER, D. 2012. Identifying Influential and Susceptible Members of Social Networks. *Science 337,* 6092, 337–341.

BLONDEL, V., GUILLAUME, J., LAMBIOTTE, R., AND LEFEBVRE, E. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment 2008*, P10008.

BROECHELER, M., SHAKARIAN, P., AND SUBRAHMANIAN, V. S. 2010. A scalable framework for modeling competitive diffusion in social networks. In *Proc. of SocialCom*. IEEE.

GRANOVETTER, M. 1978. Threshold models of collective behavior. *The American Journal of Sociology 83,* 6, 1420–1443.

KEMPE, D., KLEINBERG, J., AND TARDOS, E. 2003. Maximizing the spread of influence through a social network. In *Proc. of KDD '03*. ACM, 137–146.

KITSAK, M., GALLOS, L. K., HAVLIN, S., LILJEROS, F., MUCHNIK, L., STANLEY, H. E., AND MAKSE, H. A. 2010. Identification of influential spreaders in complex networks. *Nat Phys 6,* 11 (Nov.), 888–893.

LIEBERMAN, E., HAUERT, C., AND NOWAK, M. A. 2005. Evolutionary dynamics on graphs. *Nature 433,* 7023, 312–316.

SCHELLING, T. C. 1978. *Micromotives and Macrobehavior*. W.W. Norton and Co.

SEIDMAN, S. B. 1983. Network structure and minimum degree. *Social Networks 5,* 3, 269 – 287.

SHAKARIAN, P., SIMARI, G. I., AND SCHROEDER, R. 2013. MANCaLog: A logic for multi-attribute network cascades. In *Proc. of AAMAS-2013*.

SHAKARIAN, P., SUBRAHMANIAN, V. S., AND SAPINO, M. L. 2010. Using Generalized Annotated Programs to Solve Social Network Optimization Problems. In *Proc. of ICLP (tech. comm.)*.

SOOD, V., ANTAL, T., AND REDNER, S. 2008. Voter models on heterogeneous networks. *Physical Review E 77,* 4, 041121.

YANG, J. AND LESKOVEC, J. 2012. Defining and evaluating network communities based on ground-truth. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*. MDS '12. ACM, New York, NY, USA, 3:1–3:8.