MANCaLog: A Logic for Multi-Attribute Network Cascades (Extended Abstract)

Paulo Shakarian
Network Science Center and
Dept. of Electrical Engineering
and Computer Science
U.S. Military Academy
West Point, NY 10996
paulo@shakarian.net

Gerardo I. Simari
Dept. of Computer Science
University of Oxford
Wolfson Building, Parks Road
Oxford OX1 3QD, UK
gerardo.simari@cs.ox.ac.uk

Robert Schroeder
CORE Lab
Defense Analysis Dept.
Naval Postgraduate School
Monterey, CA 93943
rcschroe@nps.edu

Categories and Subject Descriptors

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Representation Languages*

General Terms

Languages, Algorithms

Keywords

Complex Networks, Cascades, Logic Programming

1. INTRODUCTION

Cascading processes on a network have been studied in a variety of disciplines, including computer science [3], biology [4], sociology [2], and economics [5]. Much existing work in this area is based on pre-existing models. However, recent examinations of social networks – both analysis of large data sets and experimental – have indicated that there may be additional factors to consider that are not taken into account by these models [1]. In this paper we introduce MANCaLog, a logical framework designed to describe cascades in complex networks that meets seven desiderata we selected based on a thorough review of the relevant literature. First, the framework must consider multiply labeled and weighted nodes and edges. This aspect is due to the fact that cascades in real-world networks do not only seem to depend on topological properties (i.e., an individual adopts a behavior after a certain number of his friends do) but also due to characteristics of that individual as well. Second, time should be explicitly represented, and (third) it should be *non-Markovian*, meaning that a node may choose to adopt or not adopt a behavior based on any previous time point (not simply the last one). Fourth, there must be some representation of uncertainty. Fifth, we must allow for competing cascades as has been previously done in the classic work of [4]. Sixth, cascades should be able to be *non-monotonic*, meaning that the number of nodes with a given property may increase *or* decrease in a given time period. Finally, such a framework should be tractable and allow for the computation of the outcome of the cascade in polynomial time. In this extended abstract, we introduce the MANCaLog language, which meets all of these properties.[1]

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013)*, Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.

2. FRAMEWORK

We assume that agents are arranged in a directed graph (or network) $G = (V, E)$, where V is the set of nodes (agents) and E the set of edges (their relationships). 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 $\mathcal{G} = V \cup E$ to denote the set of all *components* (nodes and edges) in the network.

Our logical language uses atoms, referring to labels and weights, to describe properties of the nodes and edges. The first piece of the syntax is the **network atom**. Given label $L \in \mathcal{L}$ and weight interval $bnd \subseteq [0, 1]$, then $\langle L, bnd\rangle$ is a **network atom**. An atom is fluent (resp., non-fluent) if $L \in \mathcal{L}_f$ (resp., $L \in \mathcal{L}_{nf}$). NA is the set of all possible network atoms. 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 $\emptyset$. A set of these atoms is a **world**. For a given world W, we impose the requirement that for each $L \in \mathcal{L}$ there is no more than one atom of the form $\langle L, bnd\rangle$ in W (and $bnd \neq \emptyset$). 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. The satisfaction relationship is defined as follows. If f is an atom of the form $\langle L, bnd\rangle$ then $W \models f$ iff there exists $\langle L, bnd'\rangle \in W$ s.t. $bnd' \subseteq bnd$. The satisfaction of conjunctions, disjunctions, and negations of formulas is then defined in the normal inductive manner.

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 $\langle L, [0, 1]\rangle \in W$. 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 nat-

[1]A full version of this abstract can be found at http://arxiv.org/abs/1301.0302.

ural 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 or decrease and/or become more or less certain. Next, MANCaLog facts state that some network atom is true for a node or edge during certain times. 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. Likewise, we introduce integrity constraints (ICs) as follows: 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*.

We now define MANCaLog rules. The idea behind rules is simple: an agent that meets some criteria is influenced by the set of its neighbors who possess certain properties. The amount of influence exerted on an agent 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 agent in question, (ii) specifies which of the neighbors can influence the agent, (iii) specifies the criteria that cause the agent to be influenced, and (iv) is the property of the agent 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, an influence function, which is a function $\mathit{ifl} : \mathbf{N} \times \mathbf{N} \rightarrow [0, 1] \times [0, 1]$ that satisfies the following two axioms: (1) it can be computed in PTIME and (2) for $x' > x$ we have $\mathit{ifl}(x', y) \subseteq \mathit{ifl}(x, y)$. This function takes the number of qualifying and eligible influencers and returns a bound on the new value for the weight of the property of the target node that changes. The next part of a rule is the **neighborhood criterion**: $(g_{edge}, g_{node}, h)_{\mathit{ifl}}$. Formulas g_{node} and h are non-fluent/fluent formulas that specify the (non-fluent and fluent, respectively) criteria on a given neighbor, while the non-fluent 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 criteria that an agent must satisfy in order to be influenced by its neighbors. Ideas such as "susceptibility" [1] 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" (denoted with fluent label L), is simply the label of the target agent that is influenced by its neighbors. Along with Δt, which specifies the time until the target is affected, we now have all the pieces to define a rule:

$$r = L \overset{\Delta t}{\leftarrow} f, (g_{edge}, g_{node}, h)_{\mathit{ifl}}$$

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 says that when certain conditions for an agent and its neighbors are met, the *bnd* bound for the network atom formed with label L on that agent 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 the next time step. Hence a MANCaLog 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. $\mathbf{P}$ is the set of all programs.

Our first semantic structure: the **network interpretation** is a mapping of network components to sets of network atoms, $NI : \mathcal{G} \rightarrow NA$. We will use $\mathbf{NI}$ to denote the set of all network interpretations. We 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 in the network. If a given label L does not describe a certain component c of the network, then in a valid network interpretation NI, $\langle L, [0, 1]\rangle \in NI(c)$. Now we can define a MANCaLog interpretation as a mapping of natural numbers in the interval $[0, t_{max}]$ to network interpretations, i.e., $I : \mathbf{N} \rightarrow \mathbf{NI}$. Let $\mathcal{I}$ be the set of all possible interpretations.

In the full version of the paper, we formally define what it means for an interpretation I to satisfy a program P. We then define the problems of consistency and entailment. Program P is consistent if there exists an interpretation that satisfies all elements in P. Likewise, P entails fact F iff for all models I of P, it holds that $I \models F$. In the full paper, we define an ordering over models and define the concept of minimal model – the interpretation that assigns components of the network the tightest bound possible on the weight. If we can find the minimal model, the questions of consistency and entailment can be answered easily. We prove, using a fixed-point operator (a technique common in logic programming) that the minimal model of a MANCaLog program can be found in polynomial time. Currently, we are creating an implementation for this framework and designing methods to automatically learn these programs from data.

3. ACKNOWLEDGMENTS

P.S. is supported by ARO project 2GDATXR042, and G.S. by (UK) EPSRC grant EP/J008346/1 – PrOQAW.

4. REFERENCES

[1] S. Aral and D. Walker. Identifying Influential and Susceptible Members of Social Networks. *Science*, 337(6092):337–341, 2012.

[2] M. Granovetter. Threshold models of collective behavior. *The American Journal of Sociology*, 83(6):1420–1443, 1978.

[3] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proc. of KDD '03*, pages 137–146. ACM, 2003.

[4] E. Lieberman, C. Hauert, and M. A. Nowak. Evolutionary dynamics on graphs. *Nature*, 433(7023):312–316, 2005.

[5] T. C. Schelling. *Micromotives and Macrobehavior*. W.W. Norton and Co., 1978.