

LAIR

L. Michelle P. Roos P. Shakarian G. Stoker

Department of Computer Science
Universities of Maryland

CMSC 818g - Spring 2009

LAIR Semester Goals

- 1 Implementation of a **context collector** that can successfully extract personal and social network info from Facebook to a SQL data base for representation.
- 2 Implementation of a **compelling Facebook application** used during context collection.
- 3 Establishment of a **first-order logic-based formalism** for reasoning about context over time, to include social networking information.
- 4 Implementation of a **programming interface** that can view SQL data, specifically that from the Enron Email Corpus, in our formalism described above.
- 5 Demonstrate **information visualizion techniques** on the Enron Email Corpus based on our formalism that allow the user to derive context.

Formalism

Other Work

- Ranganathan and Campbell describe a first-order logic for modeling context. However, time and probability are not explicitly part of their logic. Further, the logic does not support social network graphs.
- The field of model checking employs temporal logic. PCTL, and described by Hansson and Jonsson provide explicit is a probabilistic logic that expresses time explicitly.
- George et. al provide a framework for graphs that change over time, but apply it mainly to road networks, where the weight of edges change over time.
- Sharan and Neville provide a framework for reasoning about graphs that change over time that uses a Bayesian network. However, their approach only classifies nodes. We seek to answer predictive queries about relationships between nodes.

Our Formalism

Affiliation Network + Global Propositions + Logic



All Over Time

We propose a logic based on PCTL that represents time explicitly, relies on a **node space** and predicates to represent affiliation networks, and other predicates with free variables for location.

Formalism

pCTL is designed around a set of finite atomic propositions, denoted A . We shall explicitly describe sub-components of this set of propositions. We will classify the atomic propositions into certain categories as well as introduce predicates. When considering A , we will consider the following disjoint subsets:

$$A = G_N \cup G'_N \cup A_{user} \cup A_{global}$$

These disjoint subsets are as follows:

- G_N is the set of *static graph atoms* based on node-space N
- G'_N is the set of *dynamic graph atoms* based on node-space N
- A_{user} is the set of *user atomic propositions* not related to the graph specific to a user (i.e. atoms identifying the network a user is on)
- A_{global} is the set of *global atomic propositions* - atoms that can be true for all users (i.e. universal time)

Node Space

NODE SPACE: Let the node space be all possible nodes that can be in a graph. We shall write N for this set. As the nodes themselves are not atoms, we rely entirely on predicates for the atomic propositions relating to the graph. As we have two types of graph atoms, we have two types of graph predicates. They are as follows.

- $Pred_G$ are graph predicates that do not change. These predicates are recorded at every time interval.

Therefore, they are limited. They include the following.

- $d_edge(a, b)$ is true iff, for $a, b \in N$ there is a directed edge from node a to node b .
- $bi_dir(a, b)$ derived from d_edge , is true iff $d_edge(a, b) \wedge d_edge(b, a)$

Note that as bi_dir is derived, only d_edge needs to be stored. Hence, there are $|G_N| = |N|^2$.

- $Pred_{G'}$ are graph predicates that are specified by the user at run-time. They are user-specified as for any one of these predicates, there could be infinite atoms.
 - $path - exist(a, b, n)$ for $a, b \in N$ and natural n , returns true iff there exists a path (regardless of the direction of the edges) between a and b of length n or less.
 - $exist - complete(B)$ for $B \subseteq N$, returns true iff all the nodes in B form a complete subgraph.

Note that in the implementation, not only will the edges for the current time be recorded, but the edges for the some past amount of time, as specified by the user as well.

Structure: Markov Processes

The literature that describes pCTL uses Markov Processes as semantics. A Markov Process is essentially a Markov Decision Process that restricts each state to a single action. We will define a *Markov Process for Graphs* that includes the node-space as part of the tuple.

MARKOV PROCESS FOR GRAPHS: A *Markov Process for Graphs* (MG) $MG = \{S, A, N, s_j, P, J\}$ consists of the following

- S is a set of states
- A is the set of atomic propositions, as defined earlier
- N is the node space
- s_0 is the initial state
- $P : S \times S \rightarrow [0, 1]$ is a mapping from a state transition to a probability.
- $J : S \rightarrow 2^A$ is a mapping of states to atomic propositions.

We will often have to refer to sequences of states. We will use the following notation for the sequence of three states.

$$s_2 \rightarrow s_3 \rightarrow s_4$$

The probability of such a sequence can be calculated as follows.

$$P(s_2 \rightarrow s_3 \rightarrow s_4) = P(s_2, s_3) \cdot P(s_3, s_4)$$

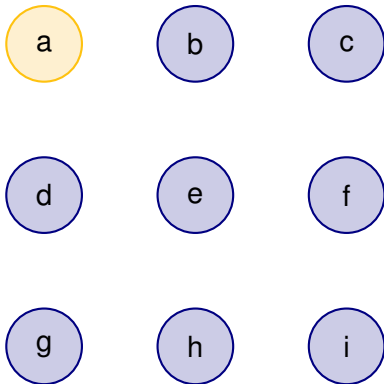
To generically represent all sequences of 2 transitions from states s_2 to s_4 , we will use the following notation.

$$s_2 \xrightarrow{2} s_4$$

Example

Who does **a** have a connection with through
 ≤ 3 people over 3 time steps?

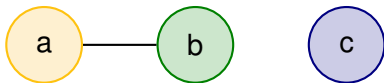
$t = 0$



Example

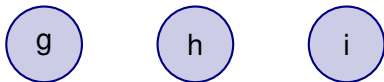
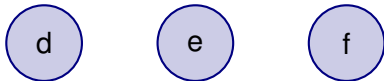
Who does **a** have a connection with through
 ≤ 3 people over 3 time steps?

$t = 1$



$Pred_G :$

$bi_dir(a,b)$



Example

Who does **a** have a connection with through
 ≤ 3 people over 3 time steps?

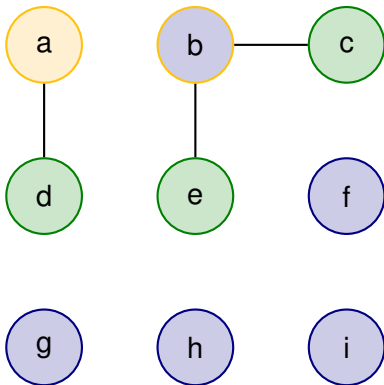
$t = 2$

$Pred_G :$

$bi_dir(a,d)$

$bi_dir(b,c)$

$bi_dir(b,e)$



Example

Who does **a** have a connection with through
 ≤ 3 people over 3 time steps?

$t = 3$

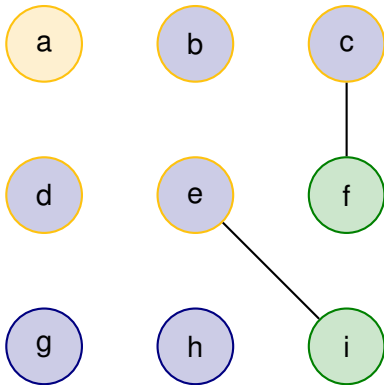
$Pred_G :$

$bi_dir(e,i)$

$bi_dir(c,f)$

Soln :

b, c, e, d, i, f



PCTL Semantics

Semantics of pCTL with Respect to MG: For the MG state s or sequence $s_0 \longrightarrow^* s'$

- $s \models_{\text{MG}} a$ iff $a \in J(s)$
- $s \models_{\text{MG}} f_1 \rightarrow f_2$ iff $s \not\models_{\text{MG}} f_1$ or $s \models_{\text{MG}} f_2$
- $s_0 \longrightarrow^* s' \models_{\text{MG}}^* f_1 U^{\leq t} f_2$ iff there exists $i \leq t$ such that $s_i \models_{\text{MG}} f_2$ and $\forall j : 0 \leq j \leq i \ s_j \models_{\text{MG}} f_1$
- $s_0 \longrightarrow^* s' \models_{\text{MG}}^* f_1 \mathcal{U}^{\leq t} f_2$ iff $s_0 \longrightarrow^* s' \models_{\text{MG}}^* f_1 U^{\leq t} f_2$ or $\forall j : 0 \leq j \leq t \ s_j \models_{\text{MG}} f_1$
- $s \models_{\text{MG}} [f]_{\geq p}$ iff for the set of finite sequences starting at s (length determined by f), $Pr_{\text{MG}}(s, \dots s_n) \geq p$ where $(s, \dots s_n)$ is the least probable sequence.
- $s \models_{\text{MG}} [f]_{> p}$ iff for the set of finite sequences starting at s (length determined by f), $Pr_{\text{MG}}(s, \dots s_n) > p$ where $(s, \dots s_n)$ is the least probable sequence (length determined by f).

PCTL Derived Operators

Additionally, several derived operators in pCTL are useful. We use a version of the derived operator "always" that includes probabilities and specifies the number of states that the formula holds true.

- Sequence Always: for all sequences, formula f is always true over the next t time periods with probability p or greater:

$$G_{\geq p}^{\leq t} f \equiv f U_{\geq p}^{\leq t} \text{FALSE}$$

- Sequence Existential: there exists a sequence where formula f is always true over the next t time periods with probability p or greater:

$$F_{\geq p}^{\leq t} f \equiv \text{TRUE} U_{\geq p}^{\leq t} f$$

- Leads-to: f_1 is followed by f_2 within t time periods with a probability of at least p :

$$f_1 \rightsquigarrow_{\geq p}^{\leq t} f_2 \equiv [G[(f_1 \rightarrow F_{\geq p}^{\leq t} f_2)]] > 1$$

Work Remaining on Formalism

- Decide if we want to have the "time window" included in the formalism or left as part of the implementation.
- Allow edge weights in predicates, decide how to implement that.
- Create a formal list of predicates to be used in implementation, including those for location.
- Decide if leads-to is adequate for rules or if we need to create something of our own.
- Do a more thorough review of related work, especially that relating to Bayesian networks.

Next Time

Status on context collector and Facebook application.