Trust Modeling and Management

Amit Konar

Math and CS, UMSL

Definitions

Trust is defined to be the firm belief in the competence of an entity to act dependably and securely within a specific context.

Distrust is defined as the firm belief in the incompetence of an entity to act dependably and securely within a specified context.

Although we define trust and distrust separately in our model, we allow the possibility of a neutral position where there is neither trust nor distrust.

Context Based Trust Model

Notations:

$(A \xrightarrow{c} B)_t$ A trusts B in context C at time t.

Usually, $(A \longrightarrow B)_t$ is a vector with 3 components: **experience, knowledge** and **recommendations.**

Components of the Trust-Relationship Vector

$$(A \xrightarrow{C} B)_t = [\, {}_A E^c_B,\ {}_A K^c_B,\ {}_S R^c_B]$$

Where

${}_A E^c_B$= magnitude of A's **experience** about B in context C,

${}_A K^c_B$= A's **knowledge** about B in context C,

${}_S R^c_B$= cumulative effect of all B's **recommendations** to A from different sources.

Range of these Parameters

Each of these factors is expressed in terms of a numeric value in the **range**

$[-1, 1] \cup \{\perp\}$.

A **negative** value for the component is used to indicate the ***trust-negative type*** of the component, whereas a **positive** value indicates a ***trust positive type*** of the component. A **zero** value indicates that the component is ***trust neutral.*** To indicate a lack of value due to insufficient information for any component, we use the special symbol $\perp$.

Modeling of the Experience Component: Strategies Used

We model experience in terms of the no. of events encountered by a truster A regarding trustee B in the context C within a specified period of time $[t_0, t_n]$.

An event can be trust-positive, trust-negative and trust-neutral depending on whether it contributes towards a trust-positive experience, a trust-negative experience or trust-neutral experience.

Intuitively, events far back in time does not count as strongly as very recent events for computing trust values. So, we need to divide the length of time into sub-intervals.

Experience Policy: Definition

An **experience policy** specifies a totally ordered set of non-overlapping intervals together with a set of non-negative weights corresponding to each element in the set of time intervals.

Recent intervals are given more weights than those far back. The whole time period $[t_0, t_n]$ is divided in such intervals, and the truster A keeps a log of events occurring in these intervals.

Computing the Experience Component

Let

P= set of all trust-positive events,

Q= set of all trust negative events

N= set of all trust-neutral events.

e_k^i denotes the k-th event in the i-th interval,

v_k^i denotes the contribution of the k-th event in the i-th interval,

$v_k^i = +1$ if e_k^i belongs to P,

$v_k^i = -1$ if e_k^i belongs to Q,

$v_k^i = 0$ if e_k^i belongs to N.

Computing Experience Component (Contd.)

The incidents I_j, corresponding to the j-th time interval is the sum of the values of all the events, trust-positive, trust-negative and trust-neutral for the given interval.

If n_j no. of events occurred in interval j, then

$$I_j = \sum_{k=1}^{n_j} v_k^j$$

The experience $_A\mathbf{E}^c{}_B = \sum_{j=1}^{n} w_j \; I_j \, / \sum_{j=1}^{n} n_j.$

For n no. of time intervals.

Computing the Knowledge Component

The knowledge component has two parts: direct knowledge and indirect knowledge (reputation).

Let

d and r be the values assigned by the truster to a trustee.

w_d and w_r be the non-negative weights for d and r respectively.

Now, the knowledge of A with regards to B for a particular context C is

$$
{}_A K^C_B = \begin{cases} d & \text{if } r = \perp \\ r & \text{if } d = \perp \\ \perp & \text{if } d = r = \perp \\ w_d\, d + w_r\, r, & \text{otherwise.} \end{cases}
$$

Here, $w_d + w_r = 1$.

Computing the Recommendation Component

Recommendation is evaluated on the basis of a recommendation value returned by a recommender to A about B. Truster A uses the "level of trust" he has on the recommender.

We define recommendation of A with regards to B for a context C

$$ {}_{S}R^{C}_{B} = \sum_{j=1}^{n} \text{Trust value of jth recommender x j-th recommender's recommendation value} $$

for n no. of recommenders.

Normalization of the Trust Vector

Basis of Normalization

Given the same values of the three factors (Exp., Know. And Reco.), two trusters may come up with two different trust values for the same trustee.

Under this circumstance we need to normalize the results by requesting all trusters to provide weights to these three components, and then taking average over each component. If the resulting averages do not satisfy linearity constraint, then divide the resulting weight by the sum of the three average weights.

The Normalization operator

Let

$W = [W_e \quad W_k \quad W_r]$ and

$T = [\,{}_A E^c_B,\ {}_A K^c_B,\ {}_S R^c_B].$

Then

$$(A \longrightarrow B)_t^N = W \circ T$$

$$= [W_e\, {}_A E^c_B \quad W_k\, {}_A K^c_B \quad W_r\, {}_S R^c_B]$$

$$\wedge \qquad\qquad \wedge \qquad\qquad \wedge$$

$$= [\,{}_A E^c_B, \quad {}_A K^c_B, \quad {}_S R^c_B].$$

Decay of Trust because of Forgetfulness of the Humans

Let

$V(T_{tn})$= decayed value of trust at time t_n.

$V(T_{ti})$ = intial value of trust at time t_i.

Then

$$V(T_{tn}) = V(\ T_{ti})\ \exp\ \{-\ (v(T_{ti})\ dt)^{2K}\}$$

where dt is $(t_n - t_i)$ and K is Boltzman's constant.

Introducing the Effect of Forgetfulness in Normalized Trust

$$\text{Trust}\ (A \xrightarrow{C} B)_{tn}^{N}$$

$$= [\ {}_{A}\hat{E}^{c}{}_{B},\ \ {}_{A}\hat{K}^{c}{}_{B},\ \ {}_{S}\hat{R}^{c}{}_{B}],\ \text{if } t_n = t_0$$

$$= \alpha\ [{}_{A}\hat{E}^{c}{}_{B},\ {}_{A}\hat{K}^{c}{}_{B},\ {}_{S}\hat{R}^{c}{}_{B}] + \beta\ [\frac{V(T^*)}{3}\ \ \frac{V(T^*)}{3}\ \ \frac{V(T^*)}{3}]$$

where

$$\alpha + \beta = 1$$

Malicious Logic

Definitions

Malicious logic is a set of instructions that cause a site's security policy to be violated.

A **Trojan horse** is a program with an *overt* (documented or known) effect and a *covert* (undocumented or unexpected) effect.

A **propagating Trojan horse** (also called a **replicating Trojan horse**) is a Trojan horse that creates a copy of itself.

Computer Virus

A **computer virus** is a program that

1) Inserts itself into one or more files, and then
2) Performs some (possibly null) actions.

The first phase is called ***insertion phase***, while the second phase is called the **execution phase**.

Pseudo-code of a Simple Computer Virus

```
Begin virus
  if spread-condition then begin
    for some set of target files do begin
      if target is not infected then begin
        determine where to place virus instructions;
        copy instructions from beginvirus to
        endvirus into target;
        alter target to execute added instructions;
      end-if;
    end-for;
  end-if;
  perform some action(s);
  goto beginning of infected program;
End virus.
```