A Reputation-Based Approach for Securing Vivaldi Embedding System

Damien Saucez Benoit Donnet Olivier Bonaventure

Université Catholique de Louvain

Eunice 2007 – Twente

http://inl.info.ucl.ac.be

July 19th, 2007

Problem

Internet applications can benefit from the ability to predict distances (RTTs). Unfortunately, direct measurements are often unattractive due to their cost.

Solution

Estimate distances without direct measurement: Model the Internet as a geometric space where hosts maintain coordinates such that $\hat{d}_{ij} \approx d_{ij}$

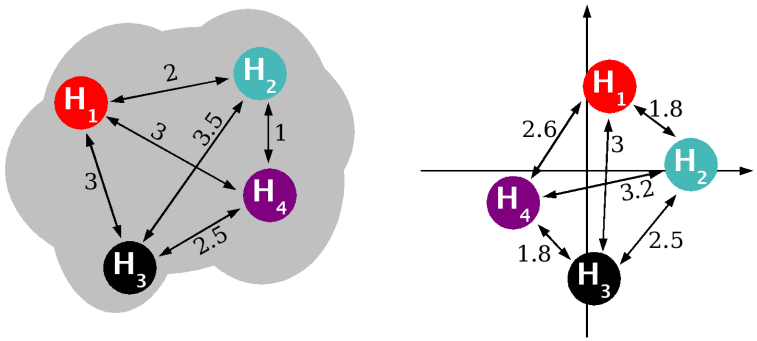


Figure: Mapping between Internet nodes and the Euclidean space

H_1 periodically updates its coordinates such that the prediction error is minimized to some neighbours based on their coordinates and the distance to them (e.g., H_2 and H_3).

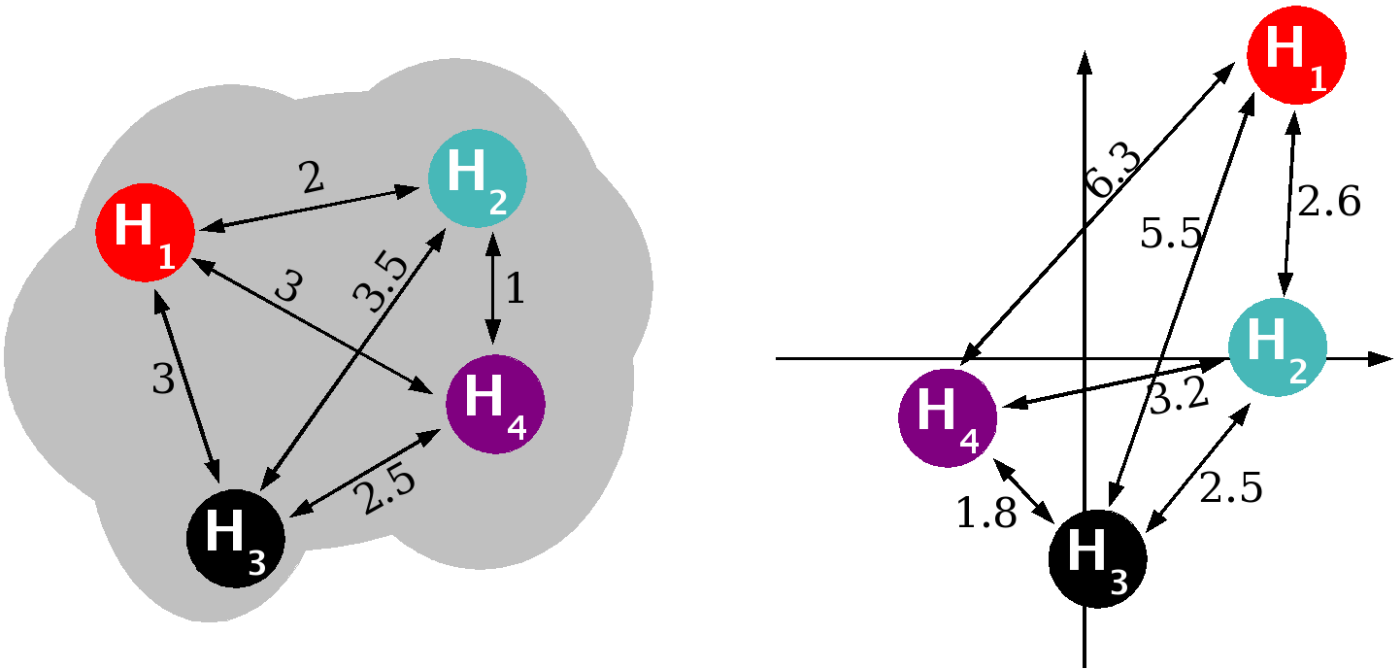


Figure: (a) Measured distances on the Internet, (b) Predicted distance in the geometric space

H_1 periodically updates its coordinates such that the prediction error is minimized to some neighbours based on their coordinates and the distance to them (e.g., H_2 and H_3).

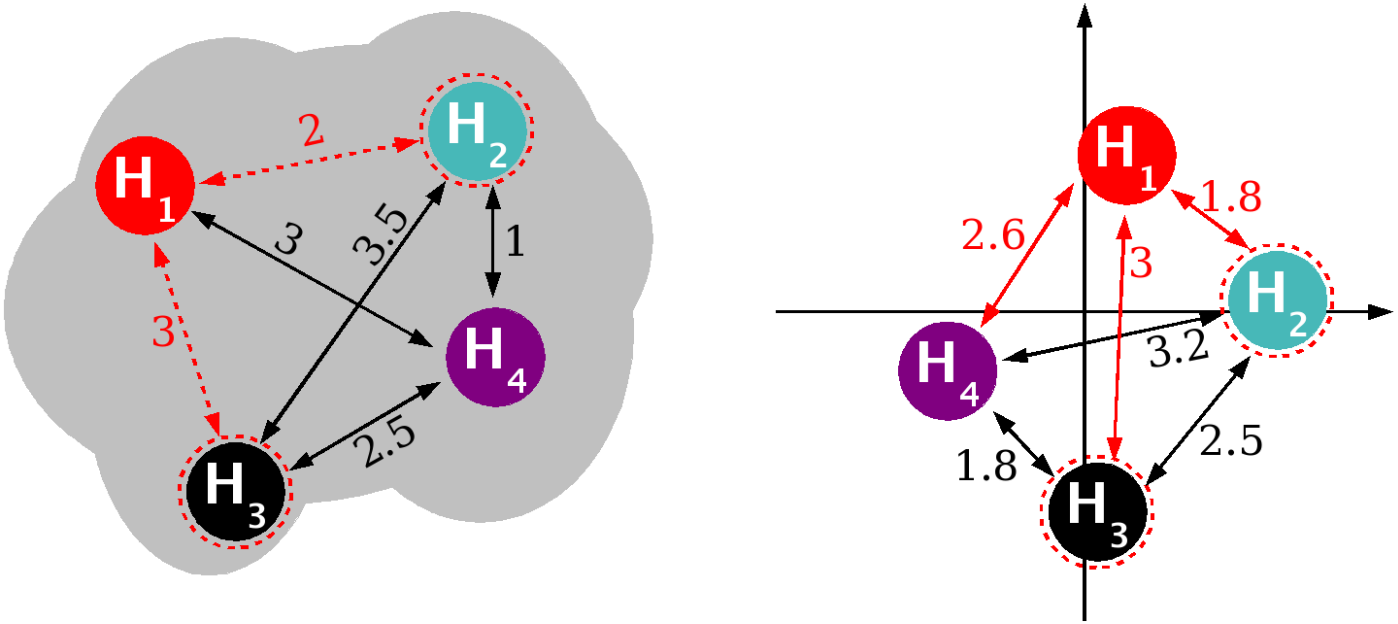


Figure: (a) Measured distances on the Internet, (b) Predicted distance in the geometric space

- Every node is an unitary mass
- A spring is placed between each pair of neighbours $\langle i, j \rangle$
- The length of the spring gives $\hat{d}_{ij}$
- The rest length of the spring is d_{ij}
- Minimizing the total energy of the masses-springs system is equivalent to minimizing the square of the prediction error [1]

The core of Vivaldi: $\vec{c}_j = \vec{c}_j + \delta \cdot \left(d_{ji} - \hat{d}_{ji} \right) \cdot u\left(\vec{c}_j - \vec{c}_i \right)$

[1] Dabek, F., Cox, R., Kaashoek, K., Morris, R.: Vivaldi, a decentralized network coordinated system. In: Proc. ACM SIGCOMM, (August 2004)

- Every node is an unitary mass
- A spring is placed between each pair of neighbours $\langle i, j \rangle$
- The length of the spring gives $\hat{d}_{ij}$
- The rest length of the spring is d_{ij}
- Minimizing the total energy of the masses-springs system is equivalent to minimizing the square of the prediction error [1]

The core of Vivaldi: $\vec{c}_j = \vec{c}_j + \delta \cdot \left(d_{ji} - \hat{d}_{ji} \right) \cdot u\left(\vec{c}_j - \vec{c}_i \right)$

[1] Dabek, F., Cox, R., Kaashoek, K., Morris, R.: Vivaldi, a decentralized network coordinated system. In: Proc. ACM SIGCOMM, (August 2004)

Figure: The spring pushes j

- Every node is an unitary mass
- A spring is placed between each pair of neighbours $\langle i, j \rangle$
- The length of the spring gives $\hat{d}_{ij}$
- The rest length of the spring is d_{ij}
- Minimizing the total energy of the masses-springs system is equivalent to minimizing the square of the prediction error [1]

The core of Vivaldi: $\vec{c}_j = \vec{c}_j + \delta \cdot \left(d_{ji} - \hat{d}_{ji} \right) \cdot u\left(\vec{c}_j - \vec{c}_i \right)$

[1] Dabek, F., Cox, R., Kaashoek, K., Morris, R.: Vivaldi, a decentralized network coordinated system. In: Proc. ACM SIGCOMM, (August 2004)

Figure: The spring reaches its rest length

Problem

Kaafar et al. showed that network coordinates systems (NCS) are sensitive to attacks [2]:
nodes rely on the information given by the neighbours to compute their own coordinates $\Rightarrow$ malicious neighbours may lie about their coordinates or RTT and alter the predictions.

Solution

Give an indication about the probability the node is malicious: **Reputation Model**

[2] Kaafar, M., Mathy, L., Turletti, T., Dabbous, W.: Virtual networks under attack: Disrupting Internet coordinate systems. In: Proc. ACM CoNEXT, (December 2006)

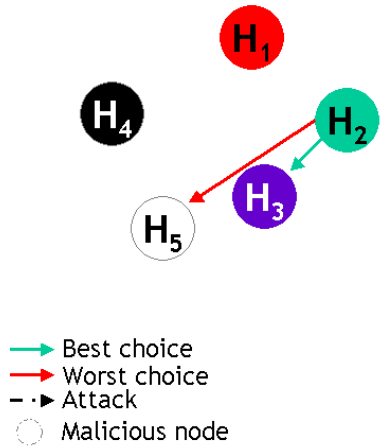
Our evaluation testbed:

- Home-made simulator;
- the King dataset with 1740 nodes;
- 32 randomly chosen neighbours at the beginning;
- 5 randomly chosen surveyors per at the beginning;
- four different attacks: constant, random, same, **repulse**;
- only one attack at a time, chosen at the beginning;
- `http://inl.info.ucl.ac.be/publications/eunice2007`.

Performance indicator:

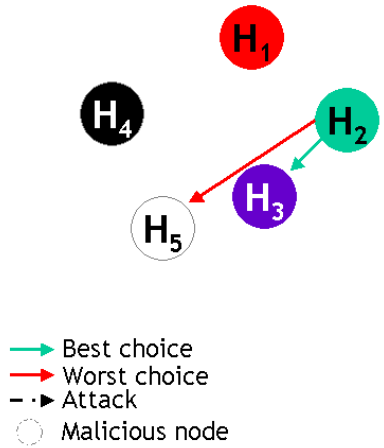
$$\text{Relative error: } e = \frac{|d_{ij} - \hat{d}_{ij}|}{d_{ij}}$$

The malicious neighbours choose coordinates where to push their target and give coordinates such that the target eventually moves to this position.

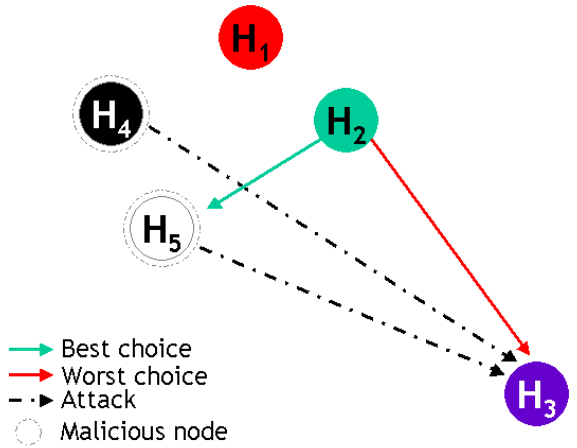


(a) Absence of attackers

The malicious neighbours choose coordinates where to push their target and give coordinates such that the target eventually moves to this position.



(c) Absence of attackers



(d) Result of an attack

After the attack, the predicted distances are inaccurate and the predictions are altered in the way the malicious nodes want.

The performances of Vivaldi rapidly decrease when the number of attackers increases. Predictions can even be worse than with randomly chosen coordinates.

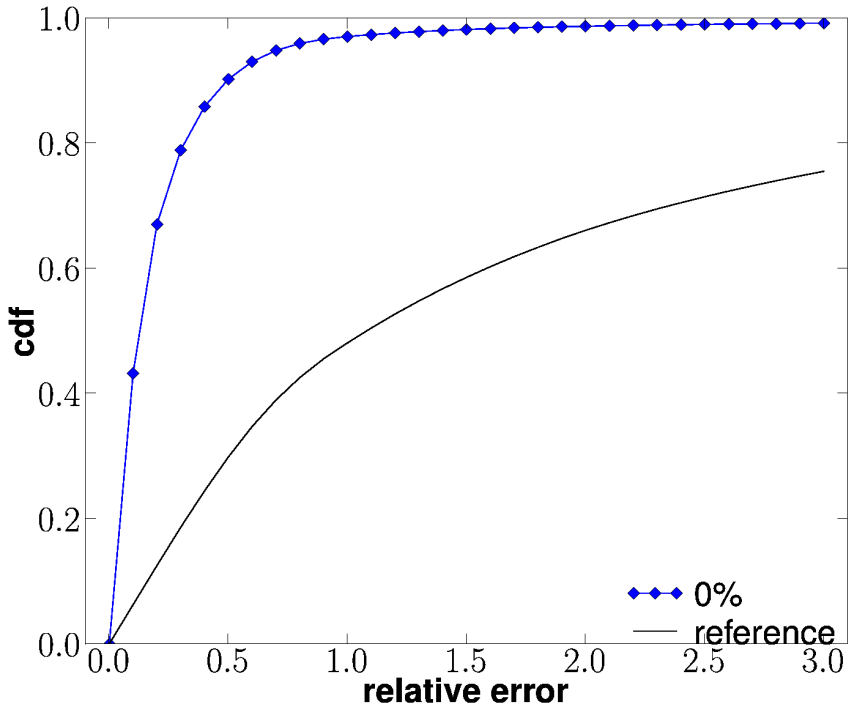


Figure: Accuracy of Vivaldi predictions in absence of attacker

The performances of Vivaldi rapidly decrease when the number of attackers increases. Predictions can even be worse than with randomly chosen coordinates.

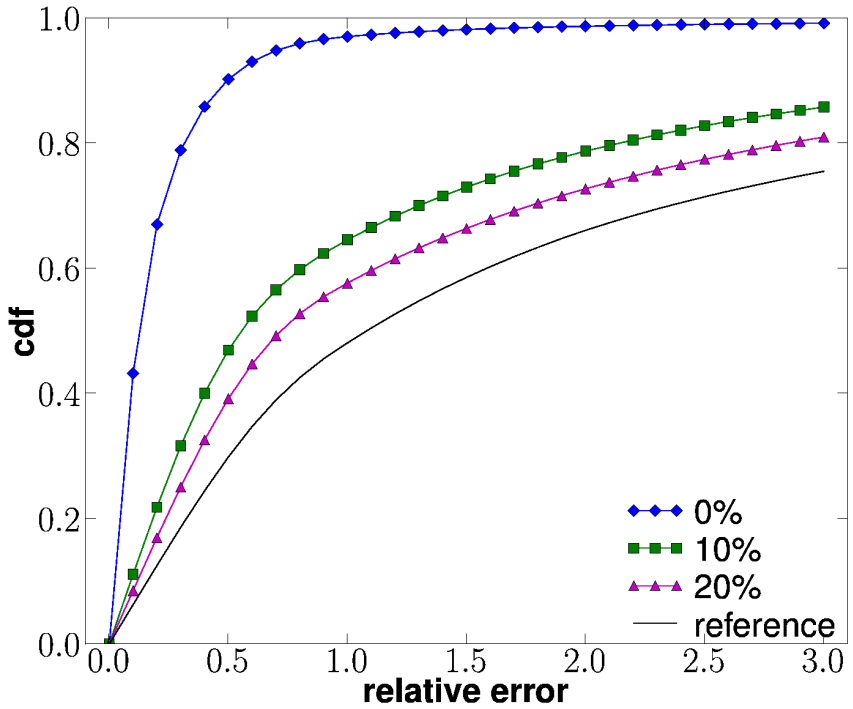


Figure: Accuracy of Vivaldi predictions in presence of few attackers

The performances of Vivaldi rapidly decrease when the number of attackers increases. Predictions can even be worse than with randomly chosen coordinates.

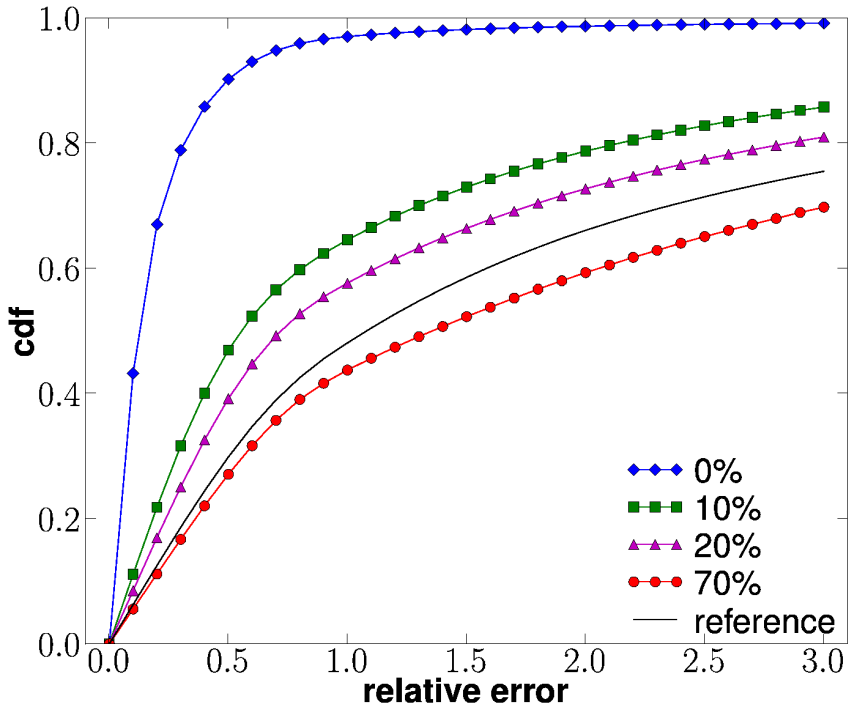


Figure: Accuracy of Vivaldi predictions in presence of many attackers

Reputation Model:
Gives an indication about the probability a node is malicious

Two new entities:

RCA: A special certification agent that gives the reputation to the nodes

Surveyors: n surveyors (set $\mathcal{S}$) are attached to each node in the system. Surveyors perform experiences measurements and trust estimation on other nodes

To compute the reputation of node A:

1. Experiences $\xi_A^{S_x}(t)$: Each surveyor measures the experience it has with A.
2. Trusts $\omega_A^{S_x}(t)$: Each surveyor computes its trust in A based on the last experiences [3].
3. Reputation $\hat{\omega}_A^{RCA}(t)$: The RCA combines the trusts and computes the reputation [3].

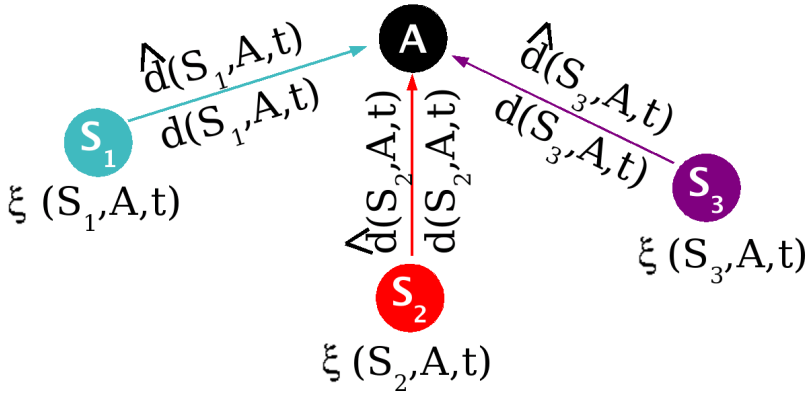


Figure: Surveyors measure the experience to A

[3] Jøsang, A.: A Logic for Uncertain Probabilities. In the International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, (June 2001)

To compute the reputation of node A:

1. Experiences $\xi_A^{S_x}(t)$: Each surveyor measures the experience it has with A.
2. Trusts $\omega_A^{S_x}(t)$: Each surveyor computes its trust in A based on the last experiences [3].
3. Reputation $\hat{\omega}_A^{RCA}(t)$: The RCA combines the trusts and computes the reputation [3].

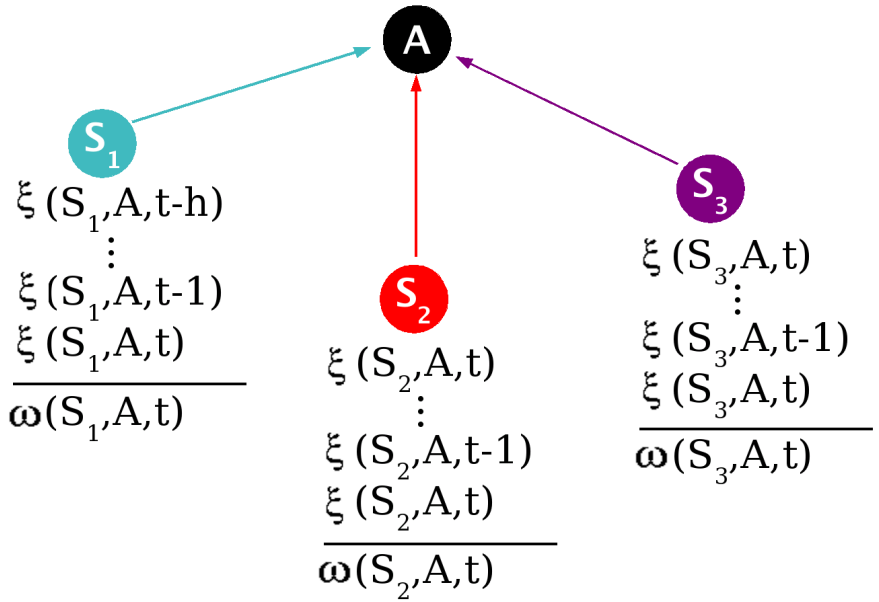


Figure: Surveyors compute the trust they have in A

[3] Jøsang, A.: A Logic for Uncertain Probabilities. In the International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, (June 2001)

To compute the reputation of node A:

1. Experiences $\xi_A^{S_x}(t)$: Each surveyor measures the experience it has with A.
2. Trusts $\omega_A^{S_x}(t)$: Each surveyor computes its trust in A based on the last experiences [3].
3. Reputation $\hat{\omega}_A^{RCA}(t)$: The RCA combines the trusts and computes the reputation [3].

[3] Jøsang, A.: A Logic for Uncertain Probabilities. In the International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, (June 2001)

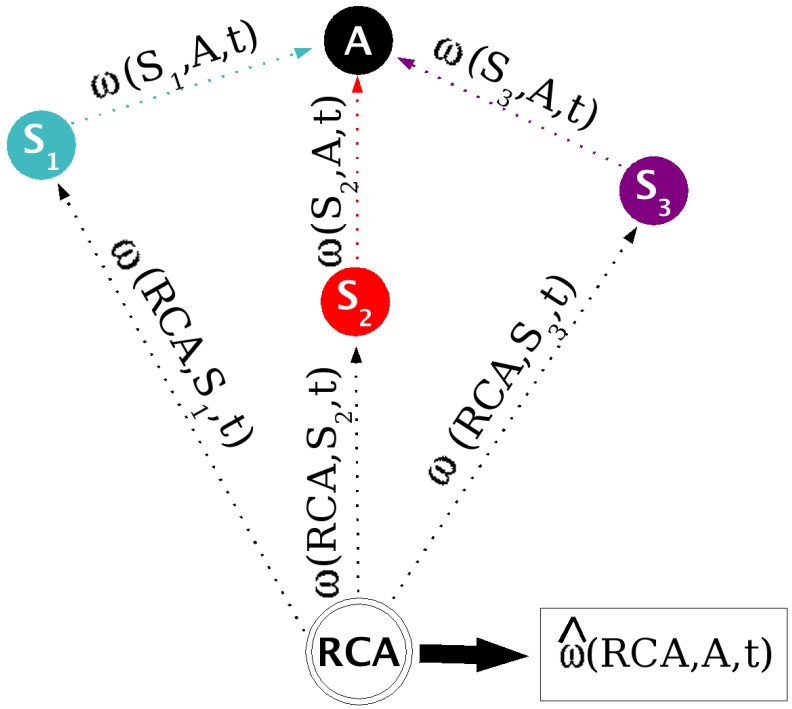


Figure: The RCA computes A's reputation based on the trusts

RVivaldi: The Reputation-based Vivaldi

The modification of j's coordinates is proportional to the reputation of its neighbour i

$$\text{Vivaldi} : \vec{c}_j = \vec{c}_j + \delta \cdot \left(d_{ji} - \hat{d}_{ji}\right) \cdot u\left(\vec{c}_j - \vec{c}_i\right)$$

$$\text{RVivaldi} : \vec{c}_j = \vec{c}_j + \hat{\varrho}_i \cdot \delta \cdot \left(d_{ji} - \hat{d}_{ji}\right) \cdot u\left(\vec{c}_j - \vec{c}_i\right)$$

In presence of attackers, the reputation approach dramatically increases the performances of Vivaldi.

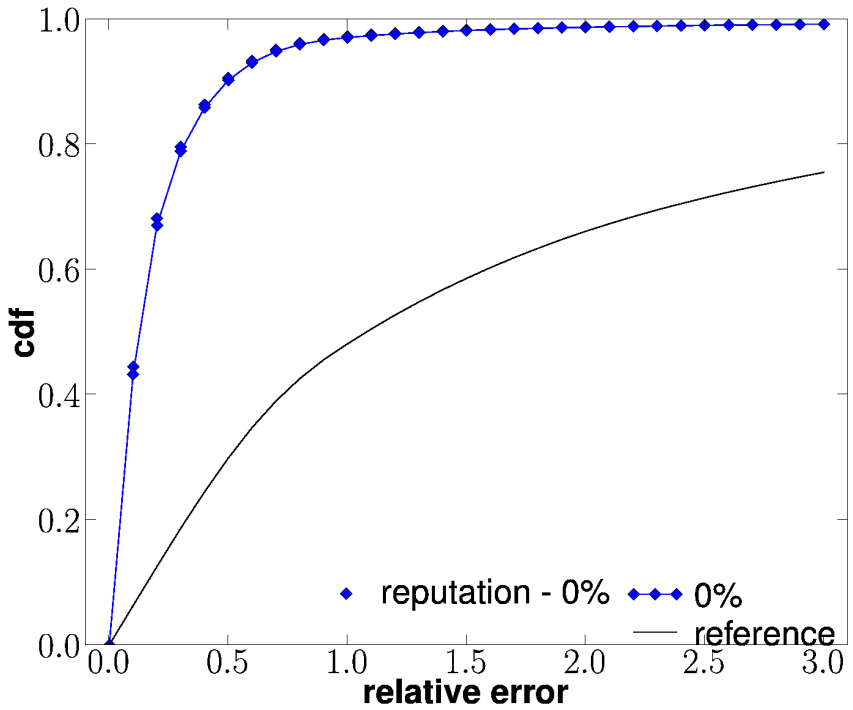


Figure: Performances comparison between Vivaldi and RVivaldi for a "repulse attack"

In presence of attackers, the reputation approach dramatically increases the performances of Vivaldi.

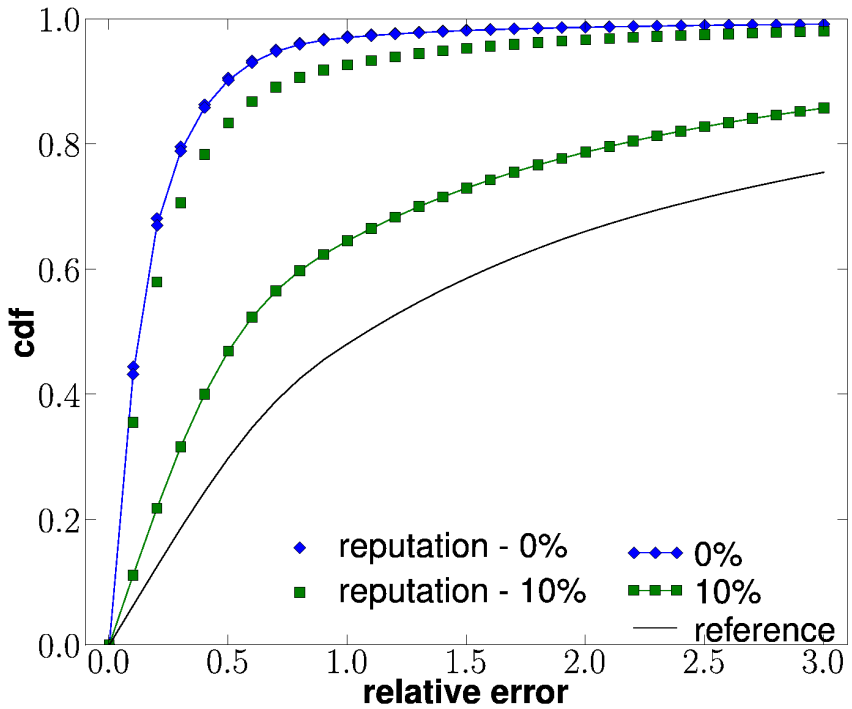


Figure: Performances comparison between Vivaldi and RVivaldi for a "repulse attack"

In presence of attackers, the reputation approach dramatically increases the performances of Vivaldi.

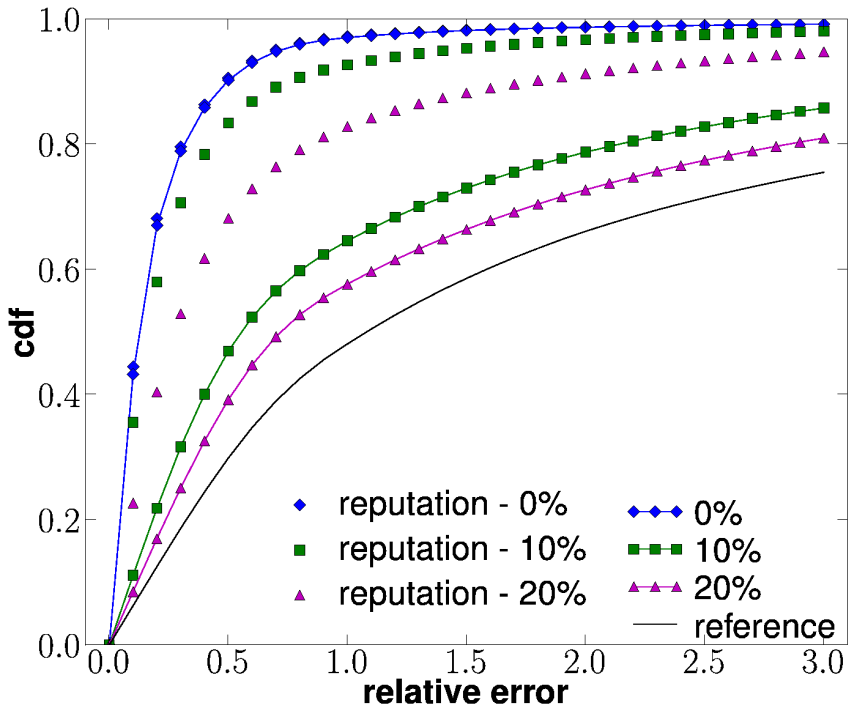


Figure: Performances comparison between Vivaldi and RVivaldi for a "repulse attack"

In presence of attackers, the reputation approach dramatically increases the performances of Vivaldi.

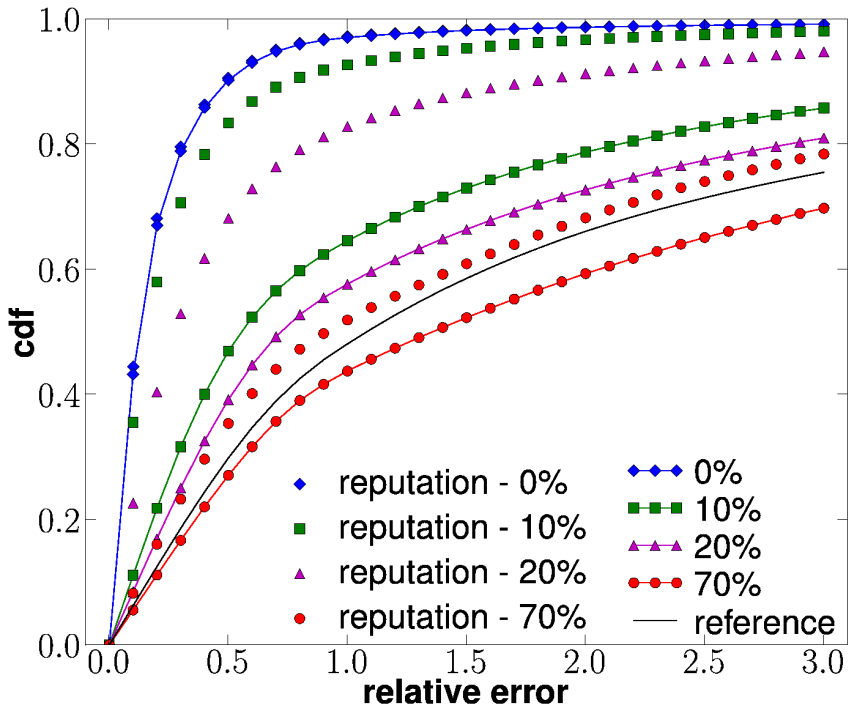


Figure: Performances comparison between Vivaldi and RVivaldi for a "repulse attack"

Current work:

- NCS permit to estimate RTT between nodes without having to contact them first
- NCS are sensitive to specific attacks on coordinates
- The reputation model estimates the probability that a node lies abouts its coordinates
- RVivaldi outperforms Vivaldi

Further works:

- Determine the best number of surveyors
- Analyze performances on other NCS in real-life environments
- **Decentralize the RCA**

Discussions

Back to you

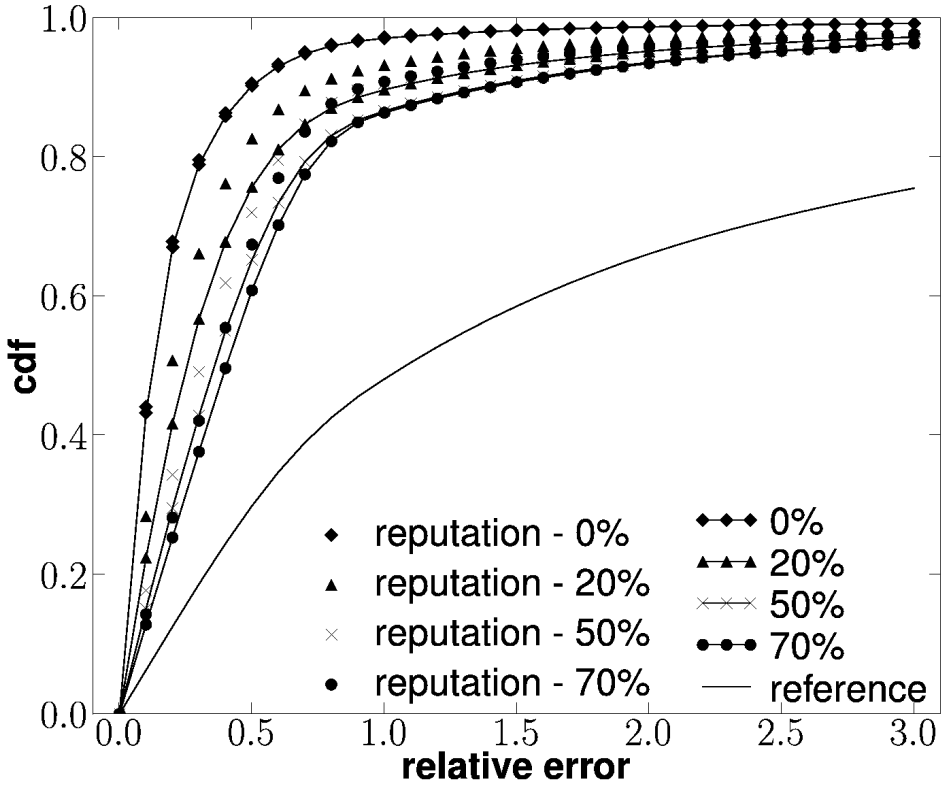


Figure: Performances comparison between Vivaldi and RVivaldi for a "Constant attack"

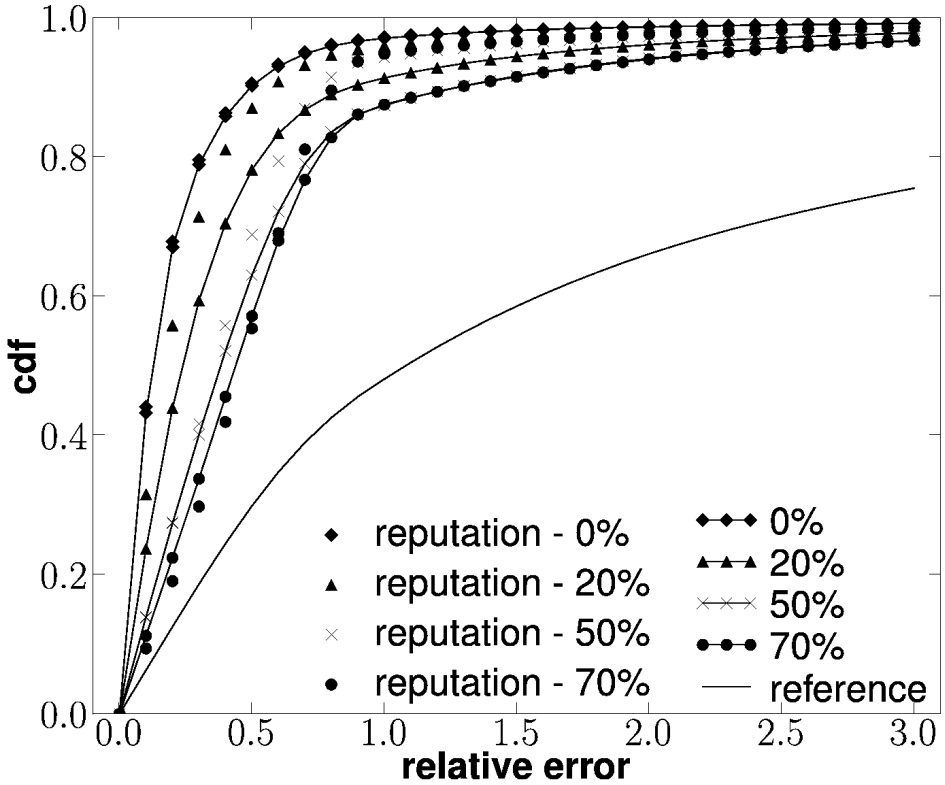


Figure: Performances comparison between Vivaldi and RVivaldi for a "Random attack"

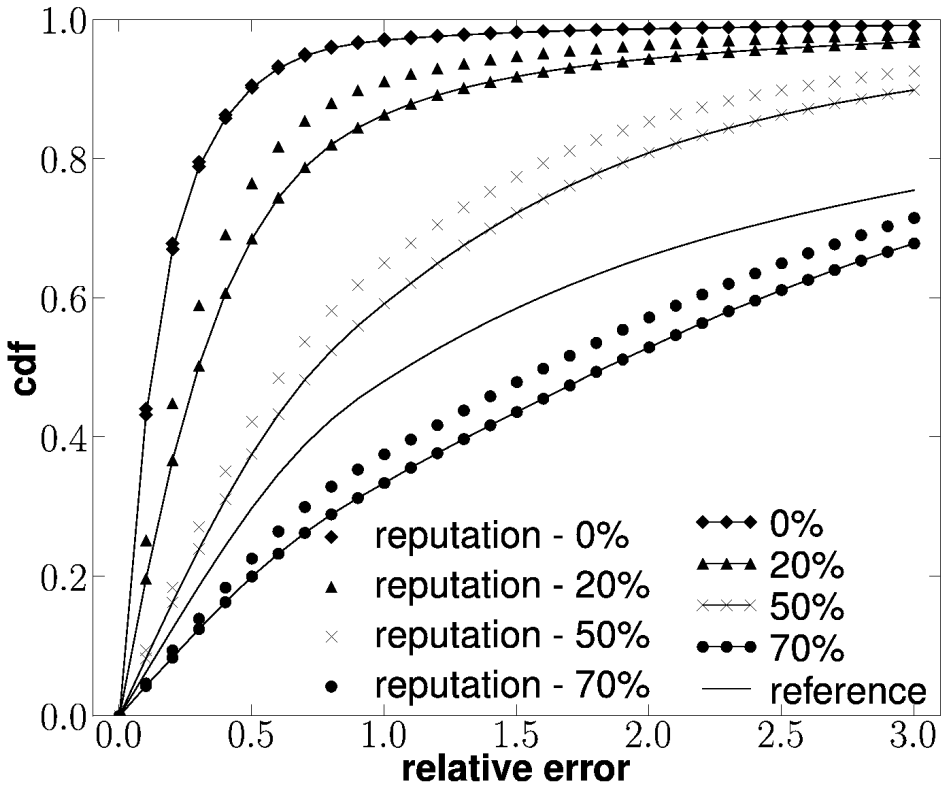


Figure: Performances comparison between Vivaldi and RVivaldi for a "Same attack"

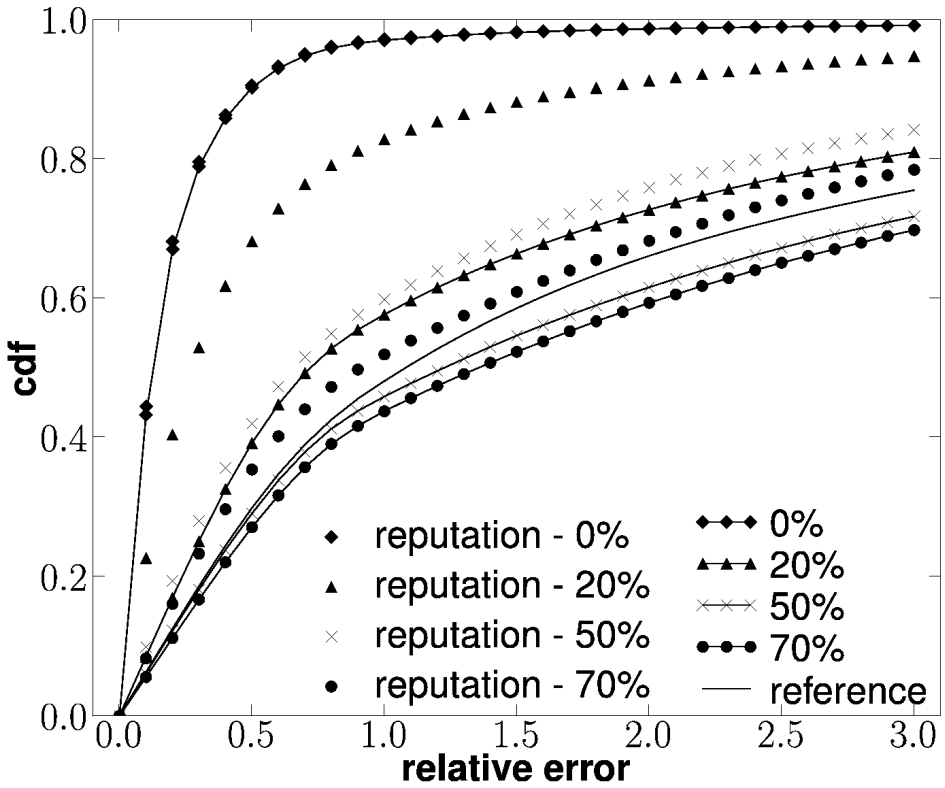


Figure: Performances comparison between Vivaldi and RVivaldi for a "Repulse attack"

$$\text{Reputation Ratio} = \frac{\text{Reputation}}{\text{No Reputation}}$$

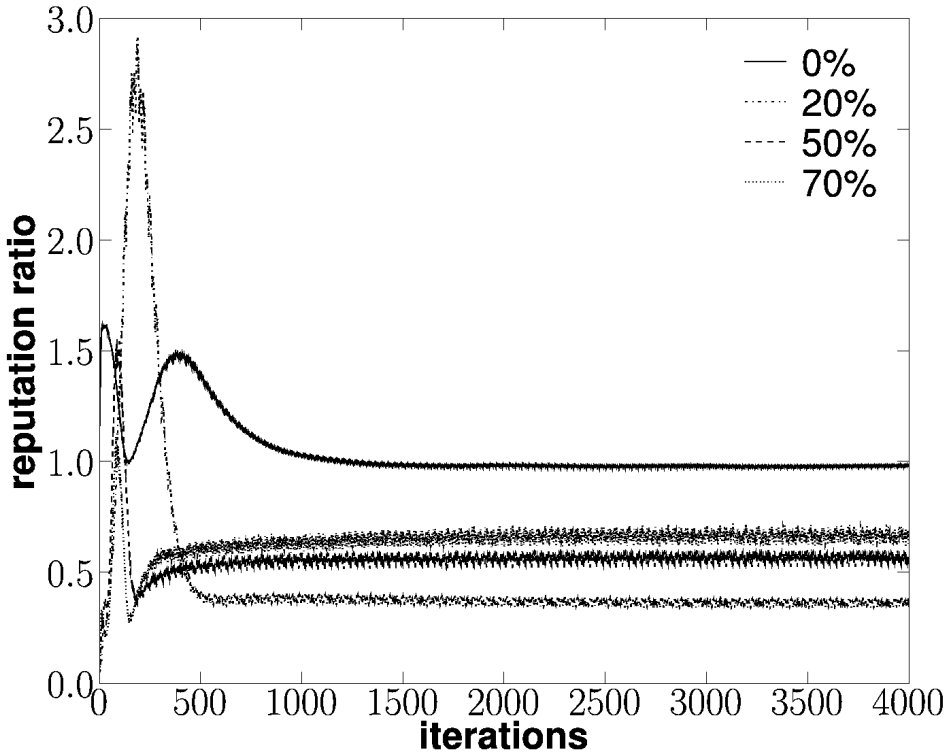


Figure: Evolution of the reputation ratio for a "Repulse attack"

$$\text{Reputation Ratio} = \frac{\text{Reputation}}{\text{No Reputation}}$$

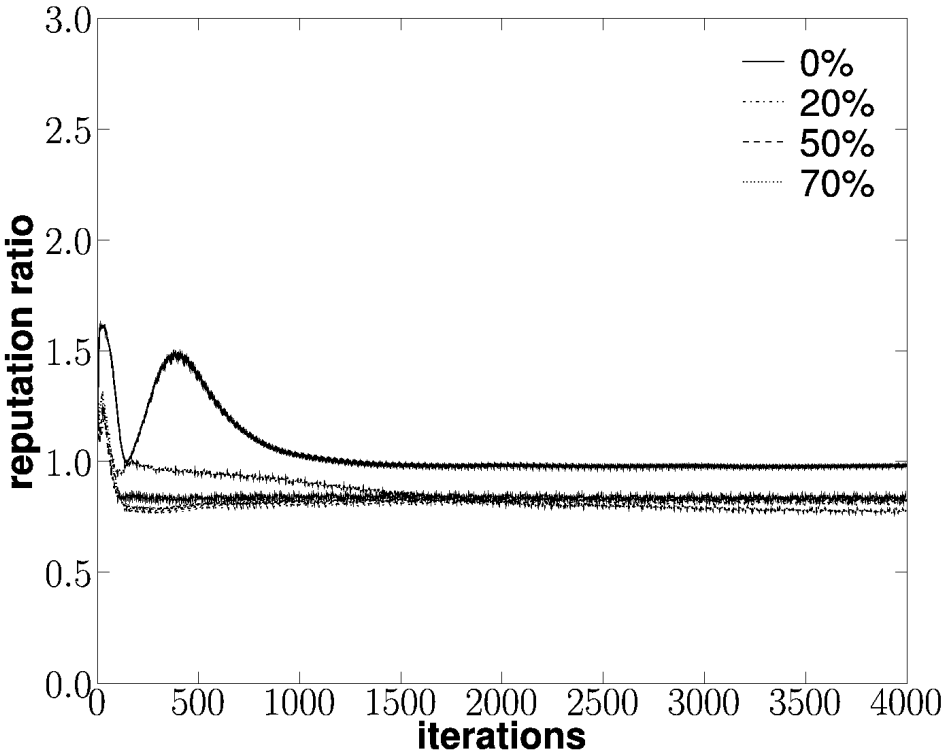


Figure: Evolution of the reputation ratio for a "Constant attack"

$$\text{Reputation Ratio} = \frac{\text{Reputation}}{\text{No Reputation}}$$

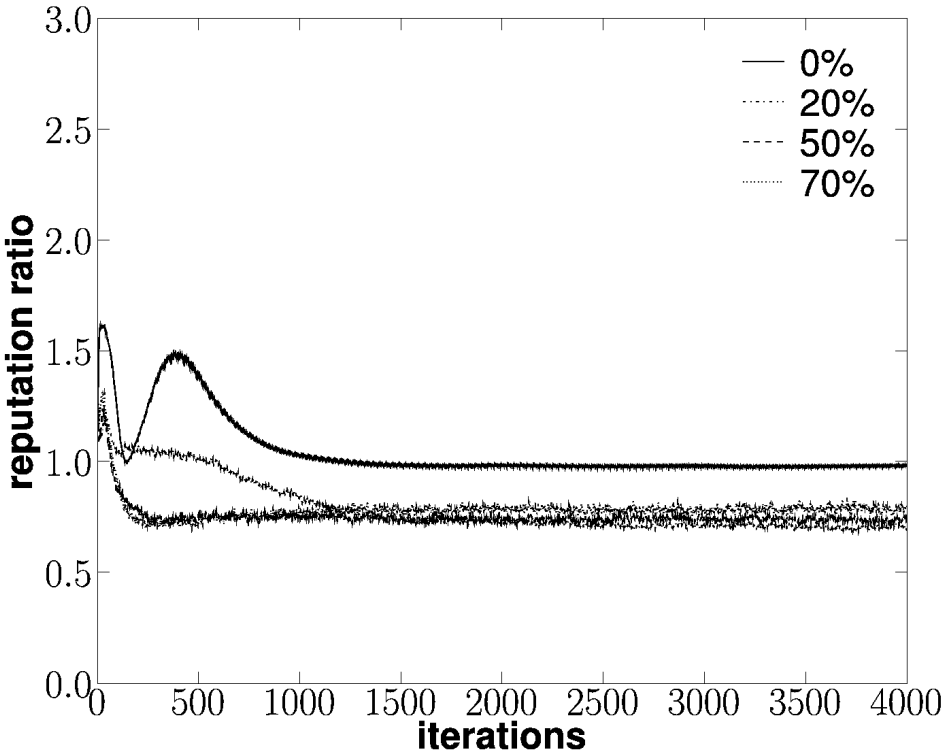


Figure: Evolution of the reputation ratio for a "Random attack"

$$\text{Reputation Ratio} = \frac{\text{Reputation}}{\text{No Reputation}}$$

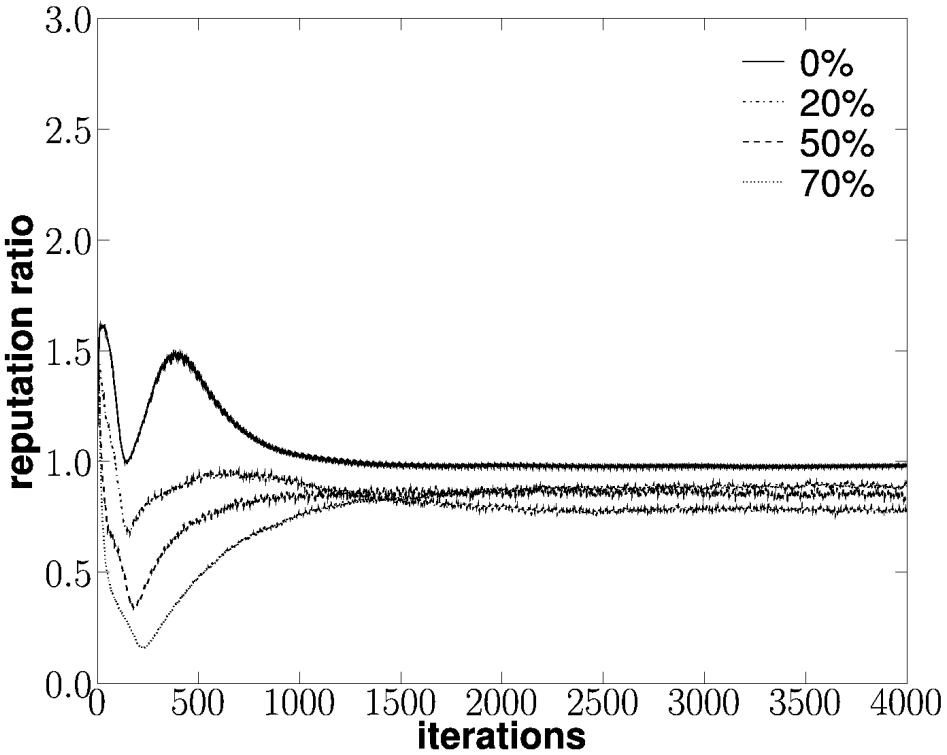


Figure: Evolution of the reputation ratio for a "Same attack"

Jøsang proposes the *uncertain probabilities model* [3]. This model is based on three fundamental functions, the *belief* $b(x)$, the *disbelief* $d(x)$ and the *uncertainty* $u(x)$. Where $b(x)$ is the total belief an observer has that x state is true, $d(x)$ that it is not true and $u(x)$ expresses the uncertainty about x.
The opinion the entity A has on x is defined as follows:

$$\omega_x^A \equiv \left(b_x^A, d_x^A, u_x^A \right). \tag{1}$$

[3] Jøsang, A.: A Logic for Uncertain Probabilities. In the International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, (June 2001)

Definition (Discounting operator $\otimes$)

If A has opinion $\omega_B^A = (b_B^A, d_B^A, u_B^A)$ on B and B has opinion $\omega_x^B = (b_x^B, d_x^B, u_x^B)$ on x, then A has the opinion $\omega_x^{AB} \equiv \omega_B^A \otimes \omega_x^B = (b_x^{AB}, d_x^{AB}, u_x^{AB})$ on x such that:

$$\begin{aligned} b_x^{AB} &= b_B^A b_x^B \\ d_x^{AB} &= b_B^A d_x^B \\ u_x^{AB} &= d_B^A + u_B^A + b_B^A u_x^B . \end{aligned} \tag{2}$$

Definition (Consensus operator $\bigoplus$)

For two different agents A and B, the opinions they have on x (ω_x^A and ω_x^B) may be different. To have a better estimate of the event's probability, the two observers may combine their observations and form a imaginary observer $[A, B]$. The consensus of $\omega_x^A = (b_x^A, d_x^A, u_x^A)$ and $\omega_x^B = (b_x^B, d_x^B, u_x^B)$ is $\omega_x^{A,B} \equiv \omega_x^A \oplus \omega_x^B = (b_x^{A,B}, d_x^{A,B}, u_x^{A,B})$ such that:

$$\begin{aligned} b_x^{A,B} &= (b_x^A u_x^B + b_x^B u_x^A)/\kappa \\ d_x^{A,B} &= (d_x^A u_x^B + d_x^B u_x^A)/\kappa \\ u_x^{A,B} &= (u_x^A u_x^B)/\kappa \end{aligned} \tag{3}$$

where $\kappa = u_x^A + u_x^B - u_x^A \cdot u_x^B$

Experience:

$$\xi(A,B,t) = 1 - \frac{\left|\hat{d}(A,B,t) - d(A,B,t)\right|}{\max\left(d(A,B,t), \hat{d}(A,B,t)\right)}. \quad (4)$$

Trustworthiness;

$$\tau(A,B,t) = a(t) \cdot \gamma \cdot \left(\sum_{i=0}^{h} (1-\gamma)^i \cdot \xi(A,B,t-i) \right). \quad (5)$$

Untrustworthiness:

$$\bar{\tau}(A,B,t) = 1 - \tau(A,B,t). \quad (6)$$

Doubt:

$$\varepsilon(A,B,t) = \sigma^2 \left(\bigcup_{i \in \{0..h\}} \xi(A,B,t-i) \right). \quad (7)$$

Trust:

$$\begin{aligned} b_B^A(t) &= \frac{\tau(A,B,t)}{(\tau(A,B,t)+\bar{\tau}(A,B,t)+\varepsilon(A,B,t))} \\ d_B^A(t) &= \frac{\bar{\tau}(A,B,t)}{(\tau(A,B,t)+\bar{\tau}(A,B,t)+\varepsilon(A,B,t))} \\ u_B^A(t) &= \frac{\varepsilon(A,B,t)}{(\tau(A,B,t)+\bar{\tau}(A,B,t)+\varepsilon(A,B,t))}. \end{aligned} \quad (8)$$

Reputation:

$$\hat{\omega}_A^{RCA} = \bigoplus_{\{H_n \in \mathcal{S}_A\}} \tilde{\omega}_{H_n}^{RCA} \otimes \hat{\omega}_A^{H_n}. \quad (9)$$

Scalar reputation:

$$\hat{\varrho}_A = \hat{b}_A^{RCA} \cdot (1 - \hat{u}_A^{RCA}). \quad (10)$$