Elements towards a foundation of computational trust

Vladimiro Sassone

ECS, University of Southampton

joint work K. Krukow and M. Nielsen

DSSE Seminar, 22 November 2006, Soton
Computational trust

Trust is an ineffable notion that permeates very many things.

What trust are we going to have in this talk?

Computer idealisation of “trust” to support decision-making in open networks. No human emotion, nor philosophical/sociological concept.

Gathering prominence in open applications involving safety guarantees in a wide sense

- **Credential-based trust**: e.g., public-key infrastructures, authentication and resource access control, network security.
- **Reputation-based trust**: e.g., social networks, P2P, trust metrics, probabilistic approaches.
- **Trust models**: e.g., security policies, languages, game theory.
- **Trust in information sources**: e.g., information filtering and provenance, content trust, user interaction, social concerns.
Computational trust

Trust is an ineffable notion that permeates very many things.

What trust are we going to have in this talk?

Computer idealisation of “trust” to support decision-making in open networks. No human emotion, nor philosophical/sociological concept.

Gathering prominence in open applications involving safety guarantees in a wide sense

- **credential-based trust**: e.g., public-key infrastructures, authentication and resource access control, network security.
- **reputation-based trust**: e.g., social networks, P2P, trust metrics, probabilistic approaches.
- **trust models**: e.g., security policies, languages, game theory.
- **trust in information sources**: e.g., information filtering and provenance, content trust, user interaction, social concerns.
Computational trust
Trust is an ineffable notion that permeates very many things.

What trust are we going to have in this talk?
Computer idealisation of “trust” to support decision-making in open networks. No human emotion, nor philosophical/sociological concept.

Gathering prominence in open applications involving safety guarantees in a wide sense

- **credential-based trust**: e.g., public-key infrastructures, authentication and resource access control, network security.

- **reputation-based trust**: e.g., social networks, P2P, trust metrics, probabilistic approaches.

- **trust models**: e.g., security policies, languages, game theory.

- **trust in information sources**: e.g., information filtering and provenance, content trust, user interaction, social concerns.
Computational trust

Trust is an ineffable notion that permeates very many things.

What trust are we going to have in this talk?

Computer idealisation of “trust” to support decision-making in open networks. No human emotion, nor philosophical/sociological concept.

Gathering prominence in open applications involving safety guarantees in a wide sense

- **credential-based trust**: e.g., public-key infrastructures, authentication and resource access control, network security.

- **reputation-based trust**: e.g., social networks, P2P, trust metrics, probabilistic approaches.

- **trust models**: e.g., security policies, languages, game theory.

- **trust in information sources**: e.g., information filtering and provenance, content trust, user interaction, social concerns.
Computational trust
Trust is an ineffable notion that permeates very many things.

What trust are we going to have in this talk?
Computer idealisation of “trust” to support decision-making in open networks. No human emotion, nor philosophical/sociological concept.

Gathering prominence in open applications involving safety guarantees in a wide sense

- **credential-based trust**: e.g., public-key infrastructures, authentication and resource access control, network security.

- **reputation-based trust**: e.g., social networks, P2P, trust metrics, probabilistic approaches.

- **trust models**: e.g., security policies, languages, game theory.

- **trust in information sources**: e.g., information filtering and provenance, content trust, user interaction, social concerns.
Trust and reputation systems

Reputation

- **behavioural**: perception that an agent creates through past actions about its intentions and norms of behaviour.
- **social**: calculated on the basis of observations made by others.

An agent’s reputation may affect the trust that others have toward it.

Trust

- **subjective**: a level of the subjective expectation an agent has about another’s future behaviour based on the history of their encounters and of hearsay.

Confidence in the trust assessment is also a parameter of importance.
Trust and reputation systems

Reputation

- **behavioural**: perception that an agent creates through past actions about its intentions and norms of behaviour.
- **social**: calculated on the basis of observations made by others.

An agent’s reputation may affect the trust that others have toward it.

Trust

- **subjective**: a level of the subjective expectation an agent has about another’s future behaviour based on the history of their encounters and of hearsay.

Confidence in the trust assessment is also a parameter of importance.
Trust and security

E.g.: Reputation-based access control

$p$'s ‘trust’ in $q$’s actions at time $t$, is determined by $p$’s observations of $q$’s behaviour up until time $t$ according to a given policy $\psi$.

Example

You download what claims to be a new cool browser from some unknown site. Your trust policy may be:

- allow the program to connect to a remote site if and only if it has neither tried to open a local file that it has not created, nor to modify a file it has created, nor to create a sub-process.
Trust and security

E.g.: Reputation-based access control

$p$’s ‘trust’ in $q$’s actions at time $t$, is determined by $p$’s observations of $q$’s behaviour up until time $t$ according to a given policy $\psi$.

Example

You download what claims to be a new cool browser from some unknown site. Your trust policy may be:

- allow the program to connect to a remote site if and only if it has neither tried to open a local file that it has not created, nor to modify a file it has created, nor to create a sub-process.
Outline

1. Some computational trust systems
2. Towards model comparison
3. Modelling behavioural information
   - Event structures as a trust model
4. Probabilistic event structures
5. A Bayesian event model
Outline

1. Some computational trust systems

2. Towards model comparison

3. Modelling behavioural information
   - Event structures as a trust model

4. Probabilistic event structures

5. A Bayesian event model
Some novel ideas well established by now.

A set $\mathcal{P}$ of $n$ peers who interact pairwise and mutually rate the interaction either sat or unsat.

- Peer $i$ computes a local ‘trust value’ in peer $j$:
  \[ s_{ij} = \text{sat}(i, j) - \text{unsat}(i, j) \leq 0. \]

- Peer $i$ then defines a normalised measure of its local trust in $j$:
  \[ c_{ij} = \frac{s_{ij}}{\sum_j s_{ij}} \]

$[c_{ij}]$ defines a Markov chain (i.e., $\sum_j c_{ij} = 1$), with stationary distribution $(t_j)_{j \in \mathcal{P}}$. The global trust value for principal $j$ is $t_j$.

Simulations prove that EigenTrust is a smart system. Yet, no much is said formally about properties, e.g. safety guarantees and the incidence of values like $t_j$. 
EigenTrust (Kamvar et al)

- Some novel ideas well established by now.
- A set $\mathcal{P}$ of $n$ peers who interact pairwise and mutually rate the interaction either \textit{sat} or \textit{unsat}.
  - Peer $i$ computes a local ‘trust value’ in peer $j$:
    \[ s_{ij} = \text{sat}(i, j) - \text{unsat}(i, j) \geq 0. \]
  - Peer $i$ then defines a normalised measure of its local trust in $j$:
    \[ c_{ij} = \frac{s_{ij}}{\sum_j s_{ij}}. \]
- $[c_{ij}]$ defines a Markov chain (i.e., $\sum_j c_{ij} = 1$), with stationary distribution $(t_j)_{j \in \mathcal{P}}$. The global trust value for principal $j$ is $t_j$.
- Simulations prove that EigenTrust is a smart system. Yet, no much is said formally about properties, e.g. safety guarantees and the incidence of values like $t_j$. 
EigenTrust (Kamvar et al)

- Some novel ideas well established by now.
- A set \( \mathcal{P} \) of \( n \) peers who interact pairwise and mutually rate the interaction either \text{sat} or \text{unsat}.
  - Peer \( i \) computes a local ‘trust value’ in peer \( j \):
    \[
    s_{ij} = \text{sat}(i, j) - \text{unsat}(i, j) \sqcup 0.
    \]
  - Peer \( i \) then defines a normalised measure of its local trust in \( j \):
    \[
    c_{ij} = \frac{s_{ij}}{\sum_j s_{ij}}
    \]
- \([c_{ij}]\) defines a Markov chain (i.e., \( \sum_j c_{ij} = 1 \)), with stationary distribution \((t_j)_{j \in \mathcal{P}}\). The global trust value for principal \( j \) is \( t_j \).
- Simulations prove that EigenTrust is a smart system. Yet, no much is said formally about properties, e.g. safety guarantees and the incidence of values like \( t_j \).
EigenTrust (Kamvar et al)

- Some novel ideas well established by now.
- A set $\mathcal{P}$ of $n$ peers who interact pairwise and mutually rate the interaction either sat or unsat.
  - Peer $i$ computes a local ‘trust value’ in peer $j$:
    \[
    s_{ij} = \text{sat}(i, j) - \text{unsat}(i, j) \sqcup 0.
    \]
  - Peer $i$ then defines a normalised measure of its local trust in $j$:
    \[
    c_{ij} = \frac{s_{ij}}{\sum_j s_{ij}}
    \]
- $[c_{ij}]$ defines a Markov chain (i.e., $\sum_j c_{ij} = 1$), with stationary distribution $(t_j)_{j \in \mathcal{P}}$. The global trust value for principal $j$ is $t_j$.
- Simulations prove that EigenTrust is a smart system. Yet, no much is said formally about properties, e.g. safety guarantees and the incidence of values like $t_j$. 
Simple Probabilistic Systems

The model $\lambda_\theta$:

- Each principal $p$ behaves in each interaction according to a fixed and independent probability $\theta_p$ of ‘success’ (and therefore $1 - \theta_p$ of ‘failure’).

The framework:

- **Interface** (Trust computation algorithm, $\mathcal{A}$):
  - **Input**: A sequence $h = x_1 x_2 \cdots x_n$ for $n \geq 0$ and $x_i \in \{s, f\}$.
  - **Output**: A probability distribution $\pi : \{s, f\} \rightarrow [0, 1]$.

- **Goal**: Output $\pi$ approximates $(\theta_p, 1 - \theta_p)$ as well as possible, under the hypothesis that input $h$ is the outcome of interactions with $p$. 
Simple Probabilistic Systems

The model $\lambda_\theta$:
- Each principal $p$ behaves in each interaction according to a fixed and independent probability $\theta_p$ of ‘success’ (and therefore $1 - \theta_p$ of ‘failure’).

The framework:
- **Interface** (Trust computation algorithm, $\mathcal{A}$):
  - **Input**: A sequence $h = x_1 x_2 \cdots x_n$ for $n \geq 0$ and $x_i \in \{s, f\}$.
  - **Output**: A probability distribution $\pi : \{s, f\} \to [0, 1]$.

- **Goal**:
  - Output $\pi$ approximates $(\theta_p, 1 - \theta_p)$ as well as possible, under the hypothesis that input $h$ is the outcome of interactions with $p$. 
Maximum likelihood (Despotovic and Aberer)

Trust computation $\mathcal{A}_0$

$$\mathcal{A}_0(s \mid h) = \frac{N_s(h)}{|h|} \quad \mathcal{A}_0(f \mid h) = \frac{N_f(h)}{|h|}$$

$N_x(h) = \text{“number of } x\text{’s in } h\text{”}$

Bayesian analysis inspired by $\lambda_\beta$ model: $f(\theta \mid \alpha \beta) \propto \theta^{\alpha-1}(1 - \theta)^{\beta-1}$

Properties:

- Well defined semantics: $\mathcal{A}_0(s \mid h)$ is interpreted as a probability of success in the next interaction.
- Solidly based on probability theory and Bayesian analysis.
- Formal result: $\mathcal{A}_0(s \mid h) \to \theta_\rho$ as $|h| \to \infty$. 
Maximum likelihood (Despotovic and Aberer)

Trust computation $A_0$

$$A_0(s \mid h) = \frac{N_s(h)}{|h|} \quad \quad A_0(f \mid h) = \frac{N_f(h)}{|h|}$$

$N_x(h) = "\text{number of } x\text{'s in } h\"$

Bayesian analysis inspired by $\lambda_\beta$ model: $f(\theta \mid \alpha \beta) \propto \theta^{\alpha-1}(1 - \theta)^{\beta-1}$

Properties:

- Well defined semantics: $A_0(s \mid h)$ is interpreted as a probability of success in the next interaction.
- Solidly based on probability theory and Bayesian analysis.
- Formal result: $A_0(s \mid h) \to \theta_\rho$ as $|h| \to \infty$. 
Beta models (Mui et al)

Even more tightly inspired by Bayesian analysis and by $\lambda_{\beta}$

Trust computation $\mathcal{A}_1$

\[
\mathcal{A}_1(s \mid h) = \frac{N_s(h) + 1}{|h| + 2} \quad \mathcal{A}_1(f \mid h) = \frac{N_f(h) + 1}{|h| + 2}
\]

$N_x(h) = \text{“number of x’s in } h\text{”}$

Properties:

- Well defined semantics: $\mathcal{A}_1(s \mid h)$ is interpreted as a probability of success in the next interaction.
- Solidly based on probability theory and Bayesian analysis.
- Formal result: Chernoff bound $\text{Prob}[\text{error} \geq \epsilon] \leq 2e^{-2m\epsilon^2}$, where $m$ is the number of trials.
Beta models (Mui et al)

Even more tightly inspired by Bayesian analysis and by $\lambda_{\beta}$

Trust computation $A_1$

\[
A_1(s \mid h) = \frac{N_s(h) + 1}{|h| + 2} \quad A_1(f \mid h) = \frac{N_f(h) + 1}{|h| + 2}
\]

$N_x(h) = \text{“number of } x\text{’s in } h\text{”}$

Properties:

- Well defined semantics: $A_1(s \mid h)$ is interpreted as a probability of success in the next interaction.
- Solidly based on probability theory and Bayesian analysis.
- Formal result: Chernoff bound $\text{Prob}[\text{error } \geq \epsilon] \leq 2e^{-2me^2}$, where $m$ is the number of trials.
Trust computation $A_2$

Based on $\lambda_\beta$, like $A_1$, but with serious approach to reputation. One of the few systems to also accounts for “malicious” reports.
Our elements of foundation

Recall the framework

- **Interface** (Trust computation algorithm, $\mathcal{A}$):
  - **Input**: A sequence $h = x_1 x_2 \cdots x_n$ for $n \geq 0$ and $x_i \in \{s, f\}$.
  - **Output**: A probability distribution $\pi : \{s, f\} \rightarrow [0, 1]$.

- **Goal**:
  - Output $\pi$ approximates $(\theta_p, 1 - \theta_p)$ as well as possible, under the hypothesis that input $h$ is the outcome of interactions with $p$.

We would like to consolidate in two directions:

1. model comparison
2. complex event model
Our elements of foundation

Recall the framework

- **Interface** (Trust computation algorithm, $\mathcal{A}$):
  - **Input**: A sequence $h = x_1 x_2 \cdots x_n$ for $n \geq 0$ and $x_i \in \{s, f\}$.
  - **Output**: A probability distribution $\pi : \{s, f\} \rightarrow [0, 1]$.

- **Goal**:
  - Output $\pi$ approximates $(\theta_p, 1 - \theta_p)$ as well as possible, under the hypothesis that input $h$ is the outcome of interactions with $p$.

We would like to consolidate in two directions:

1. model comparison
2. complex event model
Outline

1. Some computational trust systems
2. Towards model comparison
3. Modelling behavioural information
   - Event structures as a trust model
4. Probabilistic event structures
5. A Bayesian event model
Cross entropy
An information-theoretic “distance” on distributions

Cross entropy of distributions $p, q : \{o_1, \ldots, o_m\} \rightarrow [0, 1]$.

$$D(p \parallel q) = \sum_{i=1}^{m} p(o_i) \cdot \log\left(\frac{p(o_i)}{q(o_i)}\right)$$

It holds $0 \leq D(p \parallel q) \leq \infty$, and $D(p \parallel q) = 0$ iff $p = q$.

- Established measure in statistics for comparing distributions.
- Information-theoretic: the average amount of information discriminating $p$ from $q$. 
Cross entropy

An information-theoretic “distance” on distributions

Cross entropy of distributions \( p, q : \{o_1, \ldots, o_m\} \rightarrow [0, 1] \).

\[
D(p || q) = \sum_{i=1}^{m} p(o_i) \cdot \log\left(\frac{p(o_i)}{q(o_i)}\right)
\]

It holds \( 0 \leq D(p || q) \leq \infty \), and \( D(p || q) = 0 \) iff \( p = q \).

- Established measure in statistics for comparing distributions.
- Information-theoretic: the average amount of information discriminating \( p \) from \( q \).
Expected cross entropy
A measure on probabilistic trust algorithms

- Goal of a probabilistic trust algorithm $A$: given a history $X$, approximate a distribution on the outcomes $O = \{o_1, \ldots, o_m\}$.
- Different histories $X$ result in different output distributions $A(\cdot | X)$.

Expected cross entropy from $\lambda$ to $A$

$$ED^n(\lambda \parallel A) = \sum_{X \in O^n} \text{Prob}(X | \lambda) \cdot D(\text{Prob}(\cdot | X \lambda) \parallel A(\cdot | X))$$
Expected cross entropy
A measure on probabilistic trust algorithms

- Goal of a probabilistic trust algorithm $\mathcal{A}$: given a history $X$, approximate a distribution on the outcomes $O = \{o_1, \ldots, o_m\}$.
- Different histories $X$ result in different output distributions $\mathcal{A}(\cdot \mid X)$.

**Expected cross entropy from $\lambda$ to $\mathcal{A}$**

$$ED^n(\lambda \mid \mathcal{A}) = \sum_{X \in O^n} Prob(X \mid \lambda) \cdot D(Prob(\cdot \mid X \lambda) \mid \mathcal{A}(\cdot \mid X))$$
Consider the beta model $\lambda_\beta$ and the algorithms $A_0$ of maximum likelihood (Despotovic et al.) and $A_1$ beta (Mui et al.).

**Theorem**

If $\theta = 0$ or $\theta = 1$ then $A_0$ computes the exact distribution, whereas $A_1$ does not. That is, for all $n > 0$ we have:

$$\text{ED}^n(\lambda_\beta \parallel A_0) = 0 < \text{ED}^n(\lambda_\beta \parallel A_1)$$

If $0 < \theta < 1$, then $\text{ED}^n(\lambda_\beta \parallel A_0) = \infty$, and $A_1$ is always better.
An application of cross entropy

Consider the beta model $\lambda_\beta$ and the algorithms $A_0$ of maximum likelihood (Despotovic et al.) and $A_1$ beta (Mui et al.).

**Theorem**

If $\theta = 0$ or $\theta = 1$ then $A_0$ computes the exact distribution, whereas $A_1$ does not. That is, for all $n > 0$ we have:

$$ED^n(\lambda_\beta \parallel A_0) = 0 < ED^n(\lambda_\beta \parallel A_1)$$

If $0 < \theta < 1$, then $ED^n(\lambda_\beta \parallel A_0) = \infty$, and $A_1$ is always better.
An application of cross entropy

A parametric algorithm $A_\epsilon$

$$A_\epsilon(s \mid h) = \frac{N_s(h) + \epsilon}{|h| + 2\epsilon}, \quad A_\epsilon(f \mid h) = \frac{N_f(h) + \epsilon}{|h| + 2\epsilon}$$

Theorem

For any $\theta \in [0, 1]$, $\theta \neq 1/2$ there exists $\bar{\epsilon} \in [0, \infty)$ that minimises $ED_n(\lambda_\beta \parallel A_\epsilon)$, simultaneously for all $n$.

Furthermore, $ED_n(\lambda_\beta \parallel A_\epsilon)$ is a decreasing function of $\epsilon$ on the interval $(0, \bar{\epsilon})$, and increasing on $(\bar{\epsilon}, \infty)$. 
An application of cross entropy

A parametric algorithm $A_\epsilon$

$$A_\epsilon(s \mid h) = \frac{N_s(h) + \epsilon}{|h| + 2\epsilon}, \quad A_\epsilon(f \mid h) = \frac{N_f(h) + \epsilon}{|h| + 2\epsilon}$$

Theorem

For any $\theta \in [0, 1]$, $\theta \neq 1/2$ there exists $\bar{\epsilon} \in [0, \infty)$ that minimises $ED^n(\lambda_\beta \mid A_\epsilon)$, simultaneously for all $n$.

Furthermore, $ED^n(\lambda_\beta \mid A_\epsilon)$ is a decreasing function of $\epsilon$ on the interval $(0, \bar{\epsilon})$, and increasing on $(\bar{\epsilon}, \infty)$. 
A parametric algorithm $A_\epsilon$

$$A_\epsilon(s \mid h) = \frac{N_s(h) + \epsilon}{|h| + 2\epsilon}, \quad A_\epsilon(f \mid h) = \frac{N_f(h) + \epsilon}{|h| + 2\epsilon}$$

Theorem

For any $\theta \in [0, 1]$, $\theta \neq 1/2$ there exists $\bar{\epsilon} \in [0, \infty)$ that minimises $ED^n(\lambda_\beta \mid A_\epsilon)$, simultaneously for all $n$.

Furthermore, $ED^n(\lambda_\beta \mid A_\epsilon)$ is a decreasing function of $\epsilon$ on the interval $(0, \bar{\epsilon})$, and increasing on $(\bar{\epsilon}, \infty)$.

That is, unless behaviour is completely unbiased, there exists a unique best $A_\epsilon$ algorithm that for all $n$ outperforms all the others.

If $\theta = 1/2$, the larger the $\epsilon$, the better.
An application of cross entropy

A parametric algorithm $\mathcal{A}_\epsilon$

$\mathcal{A}_\epsilon(s \mid h) = \frac{N_s(h) + \epsilon}{|h| + 2\epsilon}$,

$\mathcal{A}_\epsilon(f \mid h) = \frac{N_f(h) + \epsilon}{|h| + 2\epsilon}$

Theorem

For any $\theta \in [0, 1]$, $\theta \neq 1/2$ there exists $\bar{\epsilon} \in [0, \infty)$ that minimises $\text{ED}_n(\lambda_\beta \mid \mathcal{A}_\epsilon)$, simultaneously for all $n$.

Furthermore, $\text{ED}_n(\lambda_\beta \mid \mathcal{A}_\epsilon)$ is a decreasing function of $\epsilon$ on the interval $(0, \bar{\epsilon})$, and increasing on $(\bar{\epsilon}, \infty)$.

- Algorithm $\mathcal{A}_0$ is optimal for $\theta = 0$ and for $\theta = 1$.
- Algorithm $\mathcal{A}_1$ is optimal for $\theta = \frac{1}{2} \pm \frac{1}{\sqrt{12}}$. 
Outline

1. Some computational trust systems
2. Towards model comparison
3. Modelling behavioural information
   - Event structures as a trust model
4. Probabilistic event structures
5. A Bayesian event model
Interactions and protocols

- At an abstract level, entities in a distributed system interact according to protocols;
- Information about an external entity is just information about (the outcome of) a number of (past) protocol runs with that entity.

Events as model of information

- A protocol can be specified as a concurrent process, at different levels of abstractions.
- Event structures were invented to give formal semantics to truly concurrent processes, expressing “causation” and “conflict.”
A model for behavioural information

- \( ES = (E, \leq, \#) \), with \( E \) a set of events, \( \leq \) and \( \# \) relations on \( E \).

- Information about a session is a finite set of events \( x \subseteq E \), called a configuration (which is ‘conflict-free’ and ‘causally-closed’).

- Information about several interactions is a sequence of outcomes \( h = x_1 x_2 \cdots x_n \in C^*_ES \), called a history.

eBay (simplified) example:

\[ \text{confirm} \xrightarrow{} \text{time-out} \xleftarrow{}
\]
\[ \text{pay} \xrightarrow{} \text{ignore} \xleftarrow{}
\]
\[ \text{positive} \xrightarrow{} \text{neutral} \xrightarrow{} \text{negative} \]

\( e.g., \ h = \{\text{pay, confirm, pos}\} \{\text{pay, confirm, neu}\} \{\text{pay}\} \)
A model for behavioural information

- $ES = (E, \leq, \#)$, with $E$ a set of events, $\leq$ and $\#$ relations on $E$.

- Information about a session is a finite set of events $x \subseteq E$, called a configuration (which is ‘conflict-free’ and ‘causally-closed’).

- Information about several interactions is a sequence of outcomes $h = x_1 x_2 \cdots x_n \in C_{ES}^*$, called a history.

eBay (simplified) example:

- $h = \{\text{pay, confirm, pos}\} \{\text{pay, confirm, neu}\} \{\text{pay}\}$
A model for behavioural information

- $ES = (E, \leq, \#)$, with $E$ a set of events, $\leq$ and $\#$ relations on $E$.

- Information about a session is a finite set of events $x \subseteq E$, called a configuration (which is ‘conflict-free’ and ‘causally-closed’).

- Information about several interactions is a sequence of outcomes $h = x_1x_2 \cdots x_n \in C_{ES}^*$, called a history.

eBay (simplified) example:

```
confirm ~~~~ time-out

pay ~~~~~~~~~~ ignore

positive ~~~~ neutral ~ negative
```

e.g., $h = \{\text{pay, confirm, pos}\} \cap \{\text{pay, confirm, neu}\} \cap \{\text{pay}\}$
A model for behavioural information

- $ES = (E, \leq, \#)$, with $E$ a set of events, $\leq$ and $\#$ relations on $E$.
- Information about a session is a finite set of events $x \subseteq E$, called a configuration (which is ‘conflict-free’ and ‘causally-closed’).
- Information about several interactions is a sequence of outcomes $h = x_1x_2 \cdots x_n \in C^*_{ES}$, called a history.

eBay (simplified) example:

\begin{align*}
\text{confirm} \rightsquigarrow \text{time-out} \leftarrow \text{pay} \rightsquigarrow \text{ignore} \\
\rightsquigarrow \text{positive} \rightsquigarrow \text{neutral} \rightsquigarrow \text{negative}
\end{align*}

e.g., $h = \{\text{pay, confirm, pos}\} \{\text{pay, confirm, neu}\} \{\text{pay}\}$
A model for behavioural information

- \( ES = (E, \leq, \#) \), with \( E \) a set of events, \( \leq \) and \( \# \) relations on \( E \).

- Information about a session is a finite set of events \( x \subseteq E \), called a **configuration** (which is ‘conflict-free’ and ‘causally-closed’).

- Information about several interactions is a sequence of outcomes \( h = x_1 x_2 \cdots x_n \in C^*_{ES} \), called a **history**.

**eBay (simplified) example:**

\[
\text{confirm} \sim \sim \sim \sim \sim \sim \text{time-out} \\
\downarrow \quad \quad \downarrow \\
\text{pay} \sim \sim \sim \sim \sim \sim \sim \sim \text{ignore} \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad \downarrow \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \downarrow \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \downarrow \\
\text{positive} \sim \sim \text{neutral} \sim \text{negative}
\]

**e.g.,** \( h = \{\text{pay, confirm, pos}\} \{\text{pay, confirm, neu}\} \{\text{pay}\} \)
Running example: interactions over an e-purse

authentic \sim forged \sim correct \sim incorrect

reject \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim \sim grant
Modelling outcomes and behaviour

- **Outcomes** are (maximal) configurations
- The e-purse example:

```
{g, a, c} {g, f, c} {g, a, i} {g, f, i}
```

```
{g, c} {g, a} {g, f} {g, i}
```

```
{r} {g}
```

- **Behaviour** is a sequence of outcomes
Modelling outcomes and behaviour

- **Outcomes** are (maximal) configurations
- The e-purse example:

  \[
  \{g,a,c\} \quad \{g,f,c\} \quad \{g,a,i\} \quad \{g,f,i\} \\
  \{g,c\} \quad \{g,a\} \quad \{g,f\} \quad \{g,i\} \\
  \{r\} \quad \{g\} \\
  \emptyset
  \]

- **Behaviour** is a sequence of outcomes
Modelling outcomes and behaviour

- **Outcomes** are (maximal) configurations
- The e-purse example:

\[
\begin{align*}
\{g, a, c\} & \quad \{g, f, c\} & \quad \{g, a, i\} & \quad \{g, f, i\} \\
\{g, c\} & \quad \{g, a\} & \quad \{g, f\} & \quad \{g, i\} \\
\{r\} & \quad \{g\} & \quad & \\
\emptyset & & & \\
\end{align*}
\]

- **Behaviour** is a sequence of outcomes
Modelling outcomes and behaviour

- **Outcomes** are (maximal) configurations
- **Behaviour** is a sequence of outcomes

The e-purse example:

\[
\begin{align*}
\{g, a, c\} & \quad \{g, f, c\} \\
\{g, c\} & \quad \{g, a\} \quad \{g, f\} \\
\{r\} & \quad \emptyset \quad \{g\} \\
\end{align*}
\]
Outline

1. Some computational trust systems
2. Towards model comparison
3. Modelling behavioural information
   - Event structures as a trust model
4. Probabilistic event structures
5. A Bayesian event model
Confusion-free event structures (Varacca et al)

- Immediate conflict $\#_\mu: e \# e'$ and there is $x$ that enables both.
- Confusion free: $\#_\mu$ is transitive and $e \#_\mu e'$ implies $[e) = [e')$.
- Cell: maximal $c \subseteq E$ such that $e, e' \in c$ implies $e \#_\mu e'$. 
Confusion-free event structures (Varacca et al)

- Immediate conflict $\#_\mu: e \not\in e'$ and there is $x$ that enables both.
- Confusion free: $\#_\mu$ is transitive and $e \not\in e'$ implies $[e] = [e']$.
- Cell: maximal $c \subseteq E$ such that $e, e' \in c$ implies $e \not\in e'$.

```
{g,a,c}  {g,f,c}  {g,a,i}  {g,f,i}
{g,c}    {g,a}    {g,f}    {g,i}
{g}      {g}      {g}      {g}
{r}      {g}      {g}      {g}
{g,a}    {g,b}    {g,c}    {g,d}
{g,c}    {g,a}    {g,b}    {g,c}
{g}      {g}      {g}      {g}
∅        {g}      {g}      {g}
```

V. Sassone (Soton)  Foundations of Computational Trust  06.11.22  24 / 35
Confusion-free event structures (Varacca et al)

- Immediate conflict $\#_\mu: e \not\equiv e'$ and there is $x$ that enables both.
- Confusion free: $\#_\mu$ is transitive and $e \not\equiv_\mu e'$ implies $[e) = [e')$.
- Cell: maximal $c \subseteq E$ such that $e, e' \in c$ implies $e \not\equiv_\mu e'$.

So, there are three cells in the e-purse event structure

Cell valuation: a function $p : E \rightarrow [0, 1]$ such that $p[c] = 1$, for all $c$. 
Cell valuation

- Authentic: 4/5
- Forged: 1/5
- Correct: 3/5
- Incorrect: 2/5

Reject: 1/4
Grant: 3/4

\[
\begin{align*}
\{g, a, c\} & \quad 9/25 \\
\{g, f, c\} & \quad 9/100 \\
\{g, a, i\} & \quad 6/25 \\
\{g, f, i\} & \quad 6/100 \\
\{g, c\} & \quad 9/20 \\
\{g, a\} & \quad 3/5 \\
\{g, f\} & \quad 3/20 \\
\{g, i\} & \quad 3/10 \\
\emptyset & \quad 1/4 \\
\{r\} & \quad 1/4 \\
\{g\} & \quad 3/4
\end{align*}
\]
Cell valuation

```
1/4 reject ~~~~~~~~~~ grant 3/4
```

```
4/5 authentic ~ forged
1/5 correct ~ incorrect
```

```
9/25 \{g,a,c\} \quad \{g,f,c\} 9/100 \quad 6/25 \{g,a,i\} \quad \{g,f,i\} 6/100
```

```
9/20 \{g,c\} \quad \{g,a\} 3/5 \quad 3/20 \{g,f\} \quad \{g,i\} 3/10
```

```
1/4 \{r\}
```

```
\emptyset 1
```

V. Sassone (Soton) Foundations of Computational Trust 06.11.22 25 / 35
Properties of cell valuations

Define \( p(x) = \prod_{e \in x} p(e) \). Then

- \( p(\emptyset) = 1 \);
- \( p(x) \geq p(x') \) if \( x \subseteq x' \);
- \( p \) is a probability distribution on maximal configurations.

So, \( p(x) \) is the probability that \( x \) is contained in the final outcome.
Outline

1. Some computational trust systems
2. Towards model comparison
3. Modelling behavioural information
   - Event structures as a trust model
4. Probabilistic event structures
5. A Bayesian event model
Estimating cell valuations

How to assign valuations to cells? They are the model’s unknowns.

Theorem (Bayes)

\[ \text{Prob}[\Theta \mid X \lambda] \propto \text{Prob}[X \mid \Theta \lambda] \cdot \text{Prob}[\Theta \mid \lambda] \]

A second-order notion: we not are interested in \( X \) or its probability, but in the expected value of \( \Theta \)! So, we will:

- start with a prior hypothesis \( \Theta \); this will be a cell valuation;
- record the events \( X \) as they happen during the interactions;
- compute the posterior; this is a new model fitting better with the evidence and allowing us better predictions (in a precise sense).

But: the posteriors need to be (interpretable as) a cell valuations.
Estimating cell valuations

How to assign valuations to cells? They are the model’s unknowns.

**Theorem (Bayes)**

\[
\text{Prob}[\Theta \mid X, \lambda] \propto \text{Prob}[X \mid \Theta, \lambda] \cdot \text{Prob}[\Theta \mid \lambda]
\]

A second-order notion: we not are interested in \(X\) or its probability, but in the expected value of \(\Theta\)! So, we will:

- start with a **prior** hypothesis \(\Theta\); this will be a cell valuation;
- record the events \(X\) as they happen during the interactions;
- compute the **posterior**; this is a new model fitting better with the evidence and allowing us better predictions (in a precise sense).

**But:** the posteriors need to be (interpretable as) a cell valuations.
Estimating cell valuations

How to assign valuations to cells? They are the model’s unknowns.

Theorem (Bayes)

\[ \text{Prob}[\Theta \mid X \lambda] \propto \text{Prob}[X \mid \Theta \lambda] \cdot \text{Prob}[\Theta \mid \lambda] \]

A second-order notion: we not are interested in \( X \) or its probability, but in the expected value of \( \Theta \)! So, we will:

- start with a prior hypothesis \( \Theta \); this will be a cell valuation;
- record the events \( X \) as they happen during the interactions;
- compute the posterior; this is a new model fitting better with the evidence and allowing us better predictions (in a precise sense).

But: the posteriors need to be (interpretable as) a cell valuations.
Estimating cell valuations

How to assign valuations to cells? They are the model’s unknowns.

Theorem (Bayes)

\[
\text{Prob}[\Theta \mid X, \lambda] \propto \text{Prob}[X \mid \Theta, \lambda] \cdot \text{Prob}[\Theta \mid \lambda]
\]

A second-order notion: we not are interested in $X$ or its probability, but in the expected value of $\Theta$! So, we will:

- start with a prior hypothesis $\Theta$; this will be a cell valuation;
- record the events $X$ as they happen during the interactions;
- compute the posterior; this is a new model fitting better with the evidence and allowing us better predictions (in a precise sense).

But: the posteriors need to be (interpretable as) a cell valuations.
Estimating cell valuations

How to assign valuations to cells? They are the model’s unknowns.

Theorem (Bayes)

\[ \text{Prob}[\Theta \mid X, \lambda] \propto \text{Prob}[X \mid \Theta, \lambda] \cdot \text{Prob}[\Theta \mid \lambda] \]

A second-order notion: we not are interested in \( X \) or its probability, but in the expected value of \( \Theta \)! So, we will:

- start with a prior hypothesis \( \Theta \); this will be a cell valuation;
- record the events \( X \) as they happen during the interactions;
- compute the posterior; this is a new model fitting better with the evidence and allowing us better predictions (in a precise sense).

But: the posteriors need to be (interpretable as) a cell valuations.
Estimating cell valuations

How to assign valuations to cells? They are the model’s unknowns.

Theorem (Bayes)

\[ \text{Prob}[\Theta | X, \lambda] \propto \text{Prob}[X | \Theta, \lambda] \cdot \text{Prob}[\Theta | \lambda] \]

A second-order notion: we are not interested in \( X \) or its probability, but in the expected value of \( \Theta \)! So, we will:

- start with a \textit{prior} hypothesis \( \Theta \); this will be a \textit{cell valuation};
- record the events \( X \) as they happen during the interactions;
- compute the \textit{posterior}; this is a new model fitting better with the evidence and allowing us better predictions (in a precise sense).

**But:** the posteriors need to be (interpretable as) a cell valuations.
Let $c_1, \ldots, c_M$ be the set of cells of $E$, with $c_i = \{e_{i1}, \ldots, e_{iK_i}\}$.

- A cell valuation assigns a distribution $\Theta_{c_i}$ to each $c_i$, the same way as an eventless model assigns a distribution $\theta$ to $\{s, f\}$.

- The occurrence of an $x$ from $\{s, f\}$ is a random process with two outcomes, a binomial (Bernoulli) trial on $\theta$.

- The occurrence of an event from cell $c_i$ is a random process with $K_i$ outcomes. That is, a multinomial trial on $\Theta_{c_i}$.

To exploit this analogy we only need to lift the $\lambda_\beta$ model to a model based on multinomial experiments.
Cells vs eventless outcomes

Let $c_1, \ldots, c_M$ be the set of cells of $E$, with $c_i = \{e_{i1}, \ldots, e_{Ki}\}$.

- A cell valuation assigns a distribution $\Theta_{c_i}$ to each $c_i$, the same way as an eventless model assigns a distribution $\theta$ to $\{s, f\}$.

- The occurrence of an $x$ from $\{s, f\}$ is a random process with two outcomes, a binomial (Bernoulli) trial on $\theta$.

- The occurrence of an event from cell $c_i$ is a random process with $K_i$ outcomes. That is, a multinomial trial on $\Theta_{c_i}$.

To exploit this analogy we only need to lift the $\lambda_\beta$ model to a model based on multinomial experiments.
Cells vs eventless outcomes

Let \( c_1, \ldots, c_M \) be the set of cells of \( E \), with \( c_i = \{ e_{i1}^i, \ldots, e_{K_i}^i \} \).

- A cell valuation assigns a distribution \( \Theta_{c_i} \) to each \( c_i \), the same way as an eventless model assigns a distribution \( \theta \) to \( \{ s, f \} \).

- The occurrence of an \( x \) from \( \{ s, f \} \) is a random process with two outcomes, a binomial (Bernoulli) trial on \( \theta \).

- The occurrence of an event from cell \( c_i \) is a random process with \( K_i \) outcomes. That is, a multinomial trial on \( \Theta_{c_i} \).

To exploit this analogy we only need to lift the \( \lambda_\beta \) model to a model based on multinomial experiments.
Cells vs eventless outcomes

Let $c_1, \ldots, c_M$ be the set of cells of $E$, with $c_i = \{e_{i1}, \ldots, e_{iK_i}\}$.

- A cell valuation assigns a distribution $\Theta_{c_i}$ to each $c_i$, the same way as an eventless model assigns a distribution $\theta$ to $\{s, f\}$.

- The occurrence of an $x$ from $\{s, f\}$ is a random process with two outcomes, a binomial (Bernoulli) trial on $\theta$.

- The occurrence of an event from cell $c_i$ is a random process with $K_i$ outcomes. That is, a multinomial trial on $\Theta_{c_i}$.

To exploit this analogy we only need to lift the $\lambda_\beta$ model to a model based on multinomial experiments.
A bit of magic: the Dirichlet probability distribution

The Dirichlet family $D(\Theta \mid \alpha) \propto \prod \Theta_1^{\alpha_1-1} \cdots \Theta_K^{\alpha_K-1}$

Theorem

The Dirichlet family is a conjugate prior for multinomial trials. That is, if

1. $Prob[\Theta \mid \lambda]$ is $D(\Theta \mid \alpha_1, \ldots, \alpha_K)$ and
2. $Prob[X \mid \Theta \lambda]$ follows the law of multinomial trials $\Theta_1^{n_1} \cdots \Theta_K^{n_K}$,

then $Prob[\Theta \mid X \lambda]$ is $D(\Theta \mid \alpha_1 + n_1, \ldots, \alpha_K + n_K)$ according to Bayes.

So, we start with a family $D(\Theta_{c_i} \mid \alpha_{c_i})$, and then use multinomial trials $X : E \to \omega$ to keep updating the valuation as $D(\Theta_{c_i} \mid \alpha_{c_i} + X_{c_i})$. 
The Dirichlet family \( D(\Theta | \alpha) \propto \prod \Theta_1^{\alpha_1 - 1} \cdots \Theta_K^{\alpha_K - 1} \)

**Theorem**

The Dirichlet family is a *conjugate prior* for multinomial trials. That is, if

- \( \text{Prob}[\Theta | \lambda] \) is \( D(\Theta | \alpha_1, ..., \alpha_K) \) and
- \( \text{Prob}[X | \Theta \lambda] \) follows the law of multinomial trials \( \Theta_1^{n_1} \cdots \Theta_K^{n_K} \),

then \( \text{Prob}[\Theta | X \lambda] \) is \( D(\Theta | \alpha_1 + n_1, ..., \alpha_K + n_K) \) according to Bayes.

So, we start with a family \( D(\Theta_{c_i} | \alpha_{c_i}) \), and then use multinomial trials \( X : E \to \omega \) to keep updating the valuation as \( D(\Theta_{c_i} | \alpha_{c_i} + X_{c_i}) \).
A bit of magic: the Dirichlet probability distribution

The Dirichlet family \( D(\Theta | \alpha) \propto \prod \Theta_1^{\alpha_1-1} \cdots \Theta_K^{\alpha_K-1} \)

**Theorem**

*The Dirichlet family is a conjugate prior for multinomial trials.* That is, if

- \( \text{Prob}[\Theta | \lambda] \text{ is } D(\Theta | \alpha_1, \ldots, \alpha_K) \text{ and} \)
- \( \text{Prob}[X | \Theta, \lambda] \text{ follows the law of multinomial trials } \Theta_1^{n_1} \cdots \Theta_K^{n_K}, \)

then \( \text{Prob}[\Theta | X, \lambda] \text{ is } D(\Theta | \alpha_1 + n_1, \ldots, \alpha_K + n_K) \text{ according to Bayes.} \)

So, we start with a family \( D(\Theta_{ci} | \alpha_{ci}) \), and then use multinomial trials \( X : E \rightarrow \omega \) to keep updating the valuation as \( D(\Theta_{ci} | \alpha_{ci} + X_{ci}). \)
The Bayesian process
Start with a uniform distribution for each cell.

\[ E[\Theta_{e_j} | X \lambda] = \frac{\alpha_{e_j} + x(e_j)}{\sum_{k=1}^{K_i} (\alpha_{e_k} + x(e_k))} \]

Corollary

\[ E[\text{next outcome is } x | X \lambda] = \prod_{e \in x} E[\Theta_e | X \lambda] \]
The Bayesian process
Suppose that \( X = \{ r \mapsto 2, g \mapsto 8, a \mapsto 7, f \mapsto 1, c \mapsto 3, i \mapsto 5 \} \). Then

\[
\begin{align*}
\text{authentic} & \sim \text{forged} \\
\text{correct} & \sim \text{incorrect} \\
\text{reject} & \sim \text{grant}
\end{align*}
\]

**Theorem**

\[
E[\Theta e_j^1 \mid X \lambda] = \frac{\alpha_{e_j^1} + X(e_j^1)}{\sum_{k=1}^{K_i} (\alpha_{e_k^j} + X(e_k^j))}
\]

**Corollary**

\[
E[\text{next outcome is } x \mid X \lambda] = \prod_{e \in x} E[\Theta e \mid X \lambda]
\]
The Bayesian process

Suppose that \( X = \{ r \mapsto 2, g \mapsto 8, a \mapsto 7, f \mapsto 1, c \mapsto 3, i \mapsto 5 \} \). Then

\[
\frac{4}{5} \quad \text{authentic} \sim \text{forged} \quad \frac{1}{5} \quad \text{correct} \sim \text{incorrect} \quad \frac{3}{5} \quad \text{grant} \quad \frac{2}{5}
\]

Theorem

\[
E[\Theta_{e_j} | X \lambda] = \frac{\alpha_{e_j} + X(e_j)}{\sum_{k=1}^{K_i} (\alpha_{e_k} + X(e_k))}
\]

Corollary

\[
E[\text{next outcome is } x | X \lambda] = \prod_{e \in x} E[\Theta_e | X \lambda]
\]
The Bayesian process

Suppose that $X = \{ r \mapsto 2, g \mapsto 8, a \mapsto 7, f \mapsto 1, c \mapsto 3, i \mapsto 5 \}$. Then

```
authentic \sim forged
```

```
correct \sim incorrect
```

```
reject \sim grant
```

**Theorem**

$$E[\Theta_{e_j} | X \lambda] = \frac{\alpha_{e_j} + X(e_j^i)}{\sum_{k=1}^{K_i} (\alpha_{e_k} + X(e_k^i))}$$

**Corollary**

$$E[\text{next outcome is } x | X \lambda] = \prod_{e \in x} E[\Theta_e | X \lambda]$$
The Bayesian process
Suppose that $X = \{r \mapsto 2, g \mapsto 8, a \mapsto 7, f \mapsto 1, c \mapsto 3, i \mapsto 5\}$. Then

\begin{align*}
&\text{4/5 authentic } \sim \text{ forged } \\
&\text{1/5 correct } \sim \text{ incorrect } \\
&\text{1/4 reject } \sim \text{ grant 3/4 }
\end{align*}

**Theorem**

\[
E[\Theta_{e_j} | X \lambda] = \frac{\alpha_{e_j} + X(e_j)}{\sum_{k=1}^{K_i} (\alpha_{e_k} + X(e_k))}
\]

**Corollary**

\[
E[\text{next outcome is } x | X \lambda] = \prod_{e \in x} E[\Theta_e | X \lambda]
\]
Interpretation of results

As a result, we have lifted the trust computational algorithms based on $\lambda_\beta$ to our event-base models by replacing $\binom{\text{Binomials (Bernoulli) trials}}{\beta\text{-distribution}} \mapsto \binom{\text{multinomial trials}}{\text{Dirichlet distribution}}.$
Future directions (1/2)

Hidden Markov Models

Probability parameters can change as the internal state change, probabilistically. HMM is $\lambda = (A, B, \pi)$, where

- $A$ is a Markov chain, describing state transitions;
- $B$ is family of distributions $B_s : O \to [0, 1]$;
- $\pi$ is the initial state distribution.

\[
\begin{align*}
\pi_1 &= 1 \\
B_1(a) &= .95 \\
B_1(b) &= .05 \\
\pi_2 &= 0 \\
B_2(a) &= .05 \\
B_2(b) &= .95
\end{align*}
\]
Future directions (1/2)

Hidden Markov Models

Probability parameters can change as the internal state change, probabilistically. HMM is $\lambda = (A, B, \pi)$, where

- $A$ is a Markov chain, describing state transitions;
- $B$ is family of distributions $B_s : O \rightarrow [0, 1]$;
- $\pi$ is the initial state distribution.

\begin{align*}
\pi_1 &= 1 \\
B_1(a) &= .95 \\
B_1(b) &= .05 \\
\pi_2 &= 0 \\
B_2(a) &= .05 \\
B_2(b) &= .95
\end{align*}

$O = \{a, b\}$
Future directions (2/2)

Hidden Markov Models

\[ \begin{align*}
\pi_1 &= 1 \\
B_1(a) &= .95 \\
B_1(b) &= .05
\end{align*} \quad O = \{a, b\} \quad \begin{align*}
\pi_2 &= 0 \\
B_2(a) &= .05 \\
B_2(b) &= .95
\end{align*} \]

Bayesian analysis:
- What models best explain (and thus predict) observations?
- How to approximate a HMM from a sequence of observations?

History \( h = a^{10} b^2 \). A counting algorithm would then assign high probability to \( a \) occurring next. But the last two \( b \)'s suggest a state change might have occurred, which would in reality make that probability very low.
Future directions (2/2)
Hidden Markov Models

\[ \begin{align*}
\pi_1 &= 1 \\
B_1(a) &= .95 \\
B_1(b) &= .05 \\
\pi_2 &= 0 \\
B_2(a) &= .05 \\
B_2(b) &= .95
\end{align*} \]

Bayesian analysis:

- What models best explain (and thus predict) observations?
- How to approximate a HMM from a sequence of observations?

History \( h = a^{10} b^2 \). A counting algorithm would then assign high probability to \( a \) occurring next. But the last two \( b \)'s suggest a state change might have occurred, which would in reality make that probability very low.
Summary

- A framework for “trust and reputation systems”
  - applications to security and history-based access control.

- Basic policies can be specified declaratively and verified efficiently. Quantified policies are expressive, and quantified model checking is decidable (though hard with many quantifiers).

- Bayesian approach to observations and approximations, formal results based on probability theory. Towards model comparison and complex-outcomes Bayesian model.

Future work

- Probabilistic logic.
- Dynamic models with variable structure.
- Better integration of reputation in the model.
- Relationships with game-theoretic models.
Summary

- A framework for “trust and reputation systems”
  - applications to security and history-based access control.

- Basic policies can be specified declaratively and verified efficiently. Quantified policies are expressive, and quantified model checking is decidable (though hard with many quantifiers).

- Bayesian approach to observations and approximations, formal results based on probability theory. Towards model comparison and complex-outcomes Bayesian model.

Future work
- Probabilistic logic.
- Dynamic models with variable structure.
- Better integration of reputation in the model.
- Relationships with game-theoretic models.