Sharing with free-riders: trust models to the rescue

Daniele Quercia, Stephen Hailes, and Licia Capra
Computer Science Department
d.quercia@cs.ucl.ac.uk

UCL

❶ Introduction

Situation - To share Internet connectivity and software, users may install appropriate applications on their mobile devices. Enthusiastic users may even tweak few lines of code to **free-ride**, i.e., make their devices exploit other devices' connectivity and software without providing anything in return. They may then tell other users how to do the same.

Problem - As tweaking instructions proliferate online, and as more users show enthusiasm for tinkering with their devices, free-riding prevails over sharing ("**Tragedy of the commons**").

Our proposal - Honest users install trust models on their mobile devices. Each trust model keeps track of which devices share and which do not. Collaborating users' devices team up. As a result, selfish users' devices are excluded.

Our research - It focuses on designing **distributed trust models**. A distributed trust model helps device *A* to decide whether to rely on device B. This decision involves **3 aspects** on which our research focuses.

How to exclude Condoleezza's device from service sharing

❷ Three aspects of our research

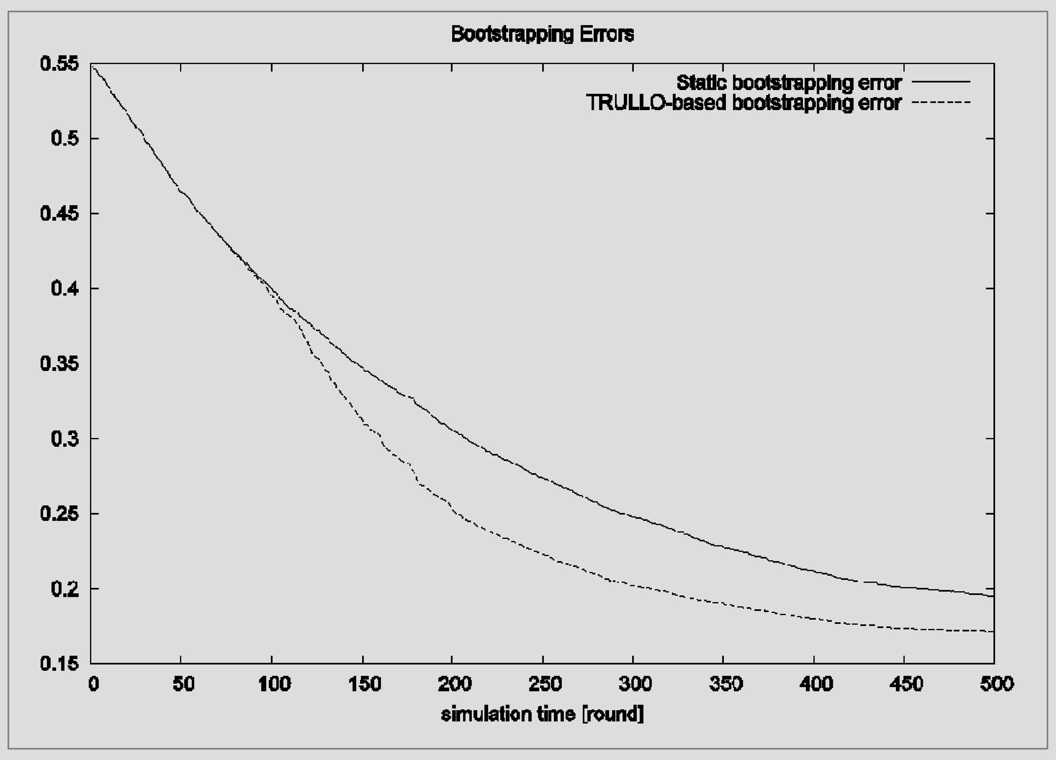
1st aspect: *A forms* its trust in *B*

Question – How does *A* set its initial trust in *B* in context *c* (e.g., bandwidth sharing)?

Existing answers – A does so:
(i) either arbitrarily (initial trust is constant);
(ii) or based on recommendations;
(iii) or close to its trust in *B* in a similar context c' (e.g., software sharing).

Our proposal – *A* uses TRULLO [1], a method that determines contextual similarity as per (iii) based on **Singular Value Decomposition** without the need for a context ontology.

Experimental Results –
A's trust model with TRULLO bootstraps closer to real trust ratings (from hostels.com) than it would do with static bootstrapping (i.e., with existing answer (i)).



Next step – Use recommendations for bootstrapping trust and deal with colluding recommenders.

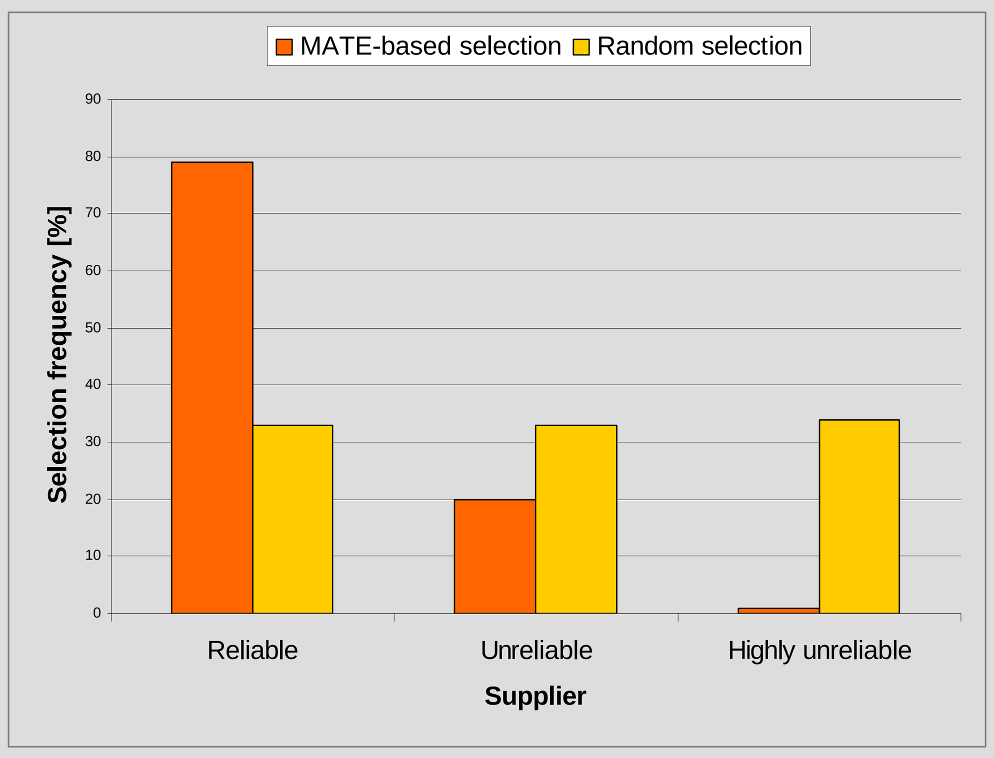
2nd aspect: *A decides* whether to rely on *B*

Question – How does *A* decide whether to rely on *B* for downloading software?

Existing answer – *A* has two available actions (rely/don't rely) and decides whether to rely on *B* or not based on *A*'s trust in *B* being above a fixed threshold.

Our proposal – *A* uses MATE [2], a **risk-aware decision model** that
(i) lists possible actions and corresponding risks;
(ii) assigns utility values to all actions;
(iii) chooses the action with the highest utility.

Experimental Results –
A's decision model downloads software mainly from reliable suppliers thus excluding unreliable ones.



Next step – Apply the decision model in contexts other than software sharing.

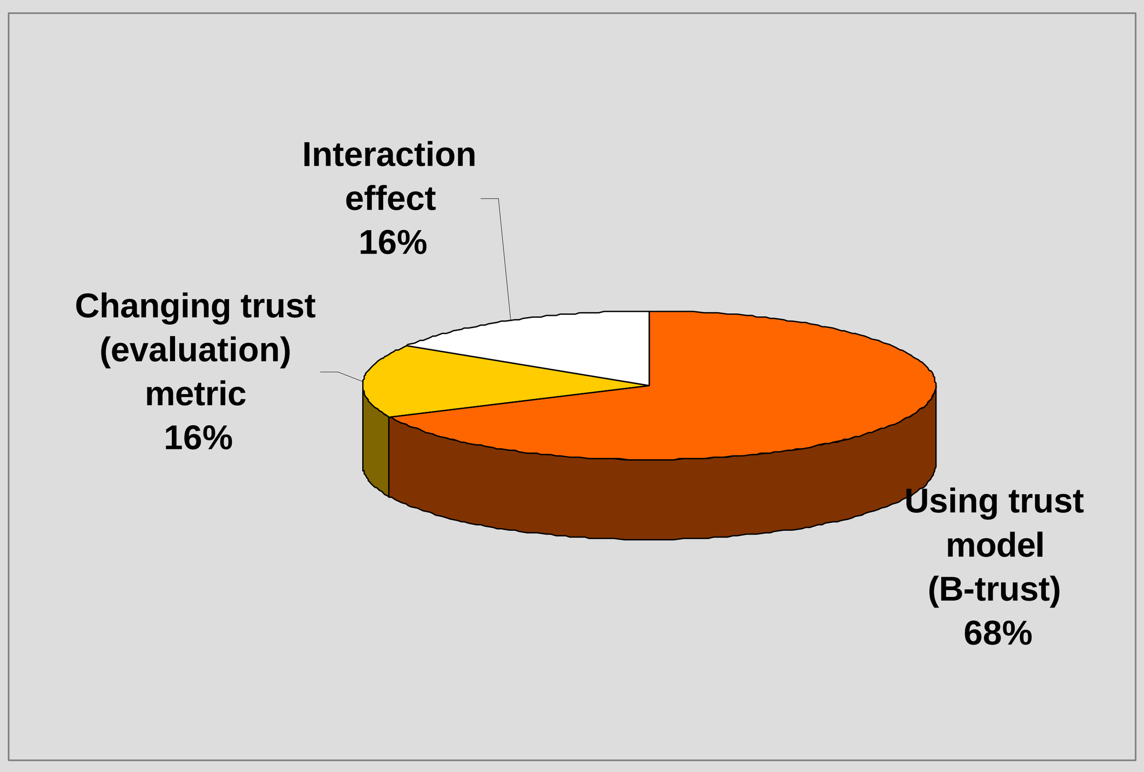
3rd aspect: *A updates* its trust in *B*

Question – How does *A* update its trust in *B* as packet forwarder after having sent Internet packets through *B*?

Existing answer – *A* decides whether the interaction has been good or bad (2-level evaluation) and consequently updates its trust with hand-crafted formulae.

Our proposal – *A* uses B-trust [3], a **trust model** that evaluates interactions at *n* levels (generally, *n*>2) and updates trust as a **Bayesian** process.

Experimental Results –
A sends more packets if it selects its next-hops with B-trust than it would do with random selection (B-trust impacts 68% on *A*'s goodput). *A* obtains even better results if B-trust switches from a binary metric (*n*=2) to a more fine-grained (*n*=4).



Next step – Look at routing in wireless mesh networks in which part of the nodes are malicious/selfish.

References

[1] D. Quercia, S. Hailes and L. Capra.
"*TRULLO: TRUst bootstrapping by Latently Lifting cOntext*". Work in progress.
[2] D. Quercia and S. Hailes.
"*MATE: Mobility and Adaptation with Trust and Expected-utility*". To appear in International Journal of Internet Technology and Secured Transactions (IJITST). 2006.
[3] D. Quercia, S. Hailes and L. Capra.
"*B-trust: Bayesian Trust Framework for Pervasive Computing*". Proceedings of the 4th International Conference on Trust Management (iTrust). LNCS. May 2006. Pisa, Italy.

Microsoft Research