The Crowds protocol

Mário S. Alvim
(msalvim@dcc.ufmg.br)

Quantitative Information Flow

DCC-UFMG
(2021/1)

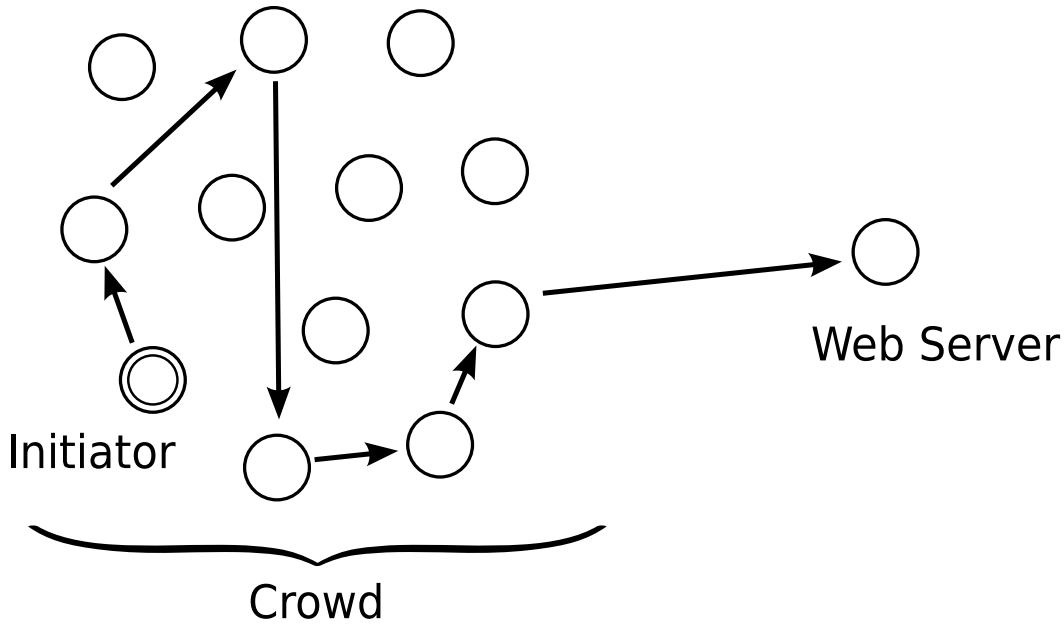
Introduction

- In this chapter we apply our theory of QIF to the analysis of the **Crowds anonymous-communication protocol**.
- The protocol is
  - simple enough to allow an analytic derivation of all relevant quantities,
  - yet rich enough to illustrate a variety of the conceptual tools we developed in our course.
- In particular, the complete analysis in our textbook involves:
  - channels,
  - hyper-distributions,
  - Bayes- and g-leakage,
  - capacity,
  - refinement, and
  - channel composition.
- In this lecture we'll go through some of our analysis.

Introduction to Crowds, and its purpose

Introduction to Crowds, and its purpose

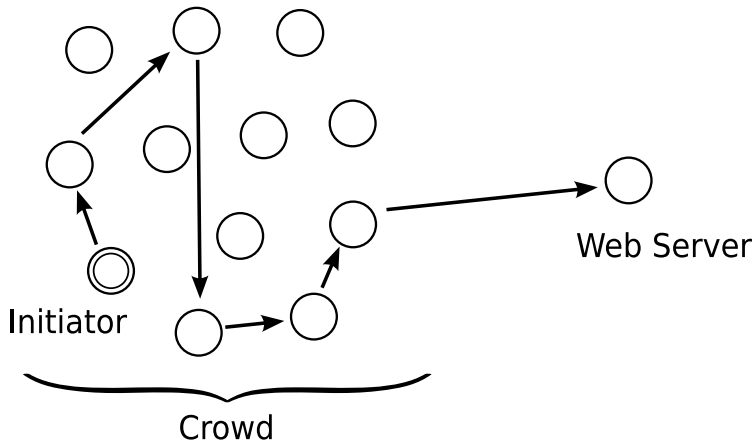
- Crowds is a simple protocol for anonymous web surfing as follows.



  - A user, called the **initiator**, wants to contact a **web server** but does not want to disclose his identity to the server.
  - The initiator achieves that by collaborating with a group of other users, called the **crowd**.

Introduction to Crowds, and its purpose

- The protocol is essentially a simple probabilistic routing protocol.



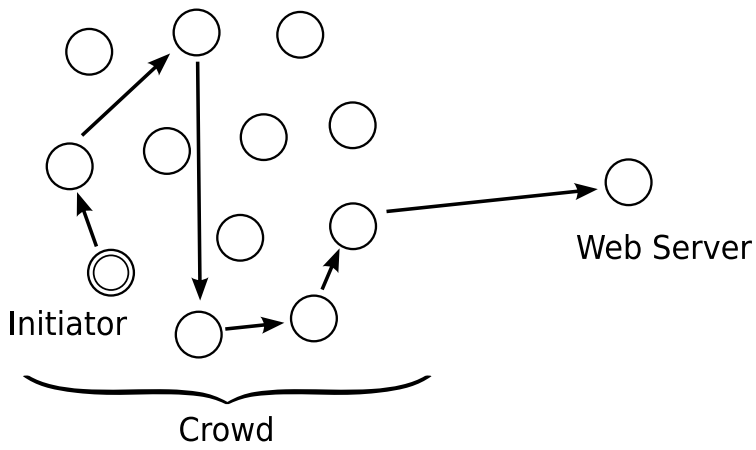
1. In the first step, the initiator:
   a) selects a user uniformly (including possibly himself), and
   b) sends his **message** to that user.

   The user who receives the message is now the (first) **forwarder**.

2. Upon receiving a message, a forwarder (whether first or subsequent) flips a (biased) coin:
   a) with probability φ he forwards the message to a new user (again chosen uniformly, including himself),
   b) but with probability $1-\varphi$ he instead delivers the message directly to the server.

Introduction to Crowds, and its purpose

- Note that the second step is itself a two-stage process:



a) first the user decides probabilistically whether to forward to another user; and

b) only then he decides probabilistically to whom.

- Those subsequent steps are called **hops**.

  Hops are distinguished from the first step because the first step cannot send to the server.

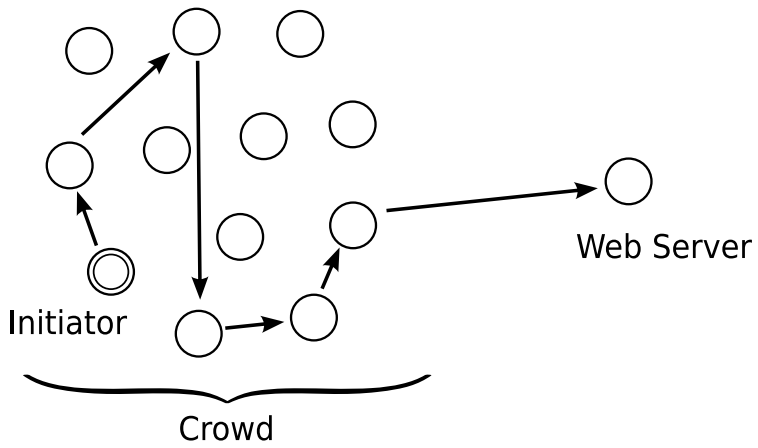
- If $\varphi < 1$, the message will with probability 1 eventually arrive at the server.

  The expected number of hops is

$$\frac{1}{1-\varphi}\,.$$

Introduction to Crowds, and its purpose

- Concerning anonymity, note the following.



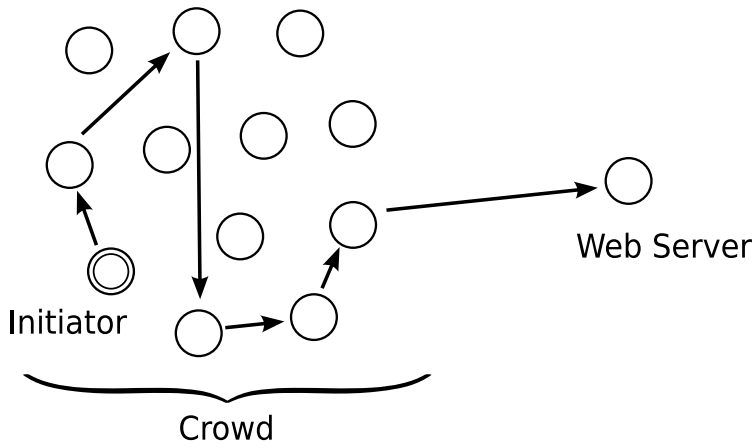
- From the point of view of the web server all users seem equally likely to have been the initiator (assuming a uniform prior).

  This is because the first step's always is a (pure) forwarding step.

- Hence, there is an intuitive notion that the initiator's identity is protected from the server.

Introduction to Crowds, and its purpose

- However the situation is more interesting (and realistic) when we assume that some users in the crowd are **corrupt**.



- If User *a* forwards a message to a **corrupted** user, then the corrupted user will report that to the adversary.

  In this case we say that User *a* was **detected**.

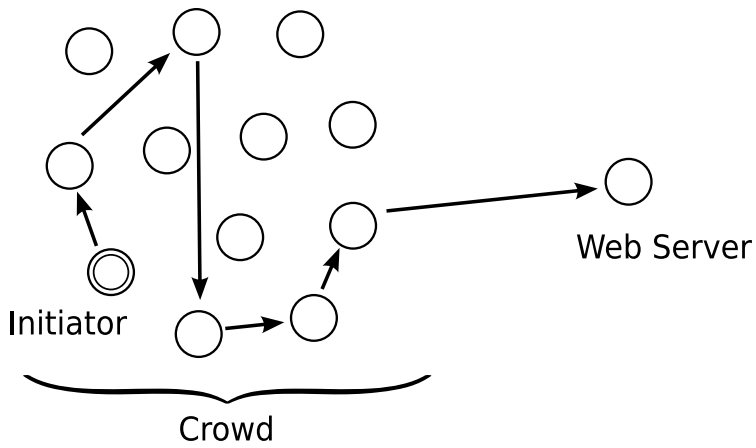
- Note that User *a*'s being detected is more likely when he is the initiator than when he is not.

  Hence the protocol does indeed leak information!

  (Detecting *a* creates a posterior where *a* has a greater chance of being the initiator than he had a priori.)

Introduction to Crowds, and its purpose

- But even if the protocol leaks information, note the following.



- If $\varphi > 0$ then the posterior is nevertheless not a point distribution.

  Hence User *a* can "plead not guilty" by claiming that he was merely forwarding for someone else.

- The greater φ is, the more plausible that claim becomes.

- Hence we can see the forwarding probability φ as trading
  - **anonymity** (the plausibility of the user's claim he's not the initiator)

    for
  - **utility** (expected number of hops).

Introduction to Crowds, and its purpose

- In their original work on Crowds, Reiter and Rubin introduced the property of "probable innocence", which intuitively requires the following.

> *"Each user appear more likely not to be the initiator of the message, than to be the initiator (although he might still appear more likely to be the initiator than any other user)."*

- That property can be formalized as follows.

Definition (18.1 – Probable innocence)

A protocol satisfies **probable innocence** iff, assuming a uniform prior over initiators, the posterior probability of any particular user's being the initiator of a message, given any observation by the adversary, is at most $1/2$.

- But how do we know if Crowds satisfies this anonymity guarantee?

  We'll use our QIF theory to do just that.

Modeling the Crowds protocol

Modeling the Crowds protocol

- To perform an analysis of the anonymity guarantees of Crowds using our theory of QIF, we start by modeling the protocol as a channnel.
- We let
  - n be the number of honest users,
  - c be the number of corrupted users, and
  - $m = n+c$ be the total number of users.
- The secret is

  "Who is the initiator?" ,

  and the set of secret values is

  $$\mathcal{X} := \{x_1, \ldots, x_n\}$$

  with x_a denoting that the initiator was (honest) User a.
- The observations are

  $$\mathcal{Y} := \{y_1, \ldots, y_n, s\} \quad , \quad \text{where}$$

  - y_b denotes that User b was detected,
  - s denotes that the message was delivered to the web server without having been forwarded to a corrupted user.

Modeling the Crowds protocol

- To construct the channel matrix $C: \mathcal{X} \rightarrowtriangle \mathcal{Y}$, we need to compute the probability of producing any y_b or s given that the secret is x_a.
- In the textbook we show (by combinatorics reasoning) the following.
  1. The probability that a message coming from initiator a reaches the server without being detected by any corrupted user (denoted by observation s) is
  $$C_{x_a,s} \quad = \quad \alpha \ := \quad \frac{n - \varphi n}{m - \varphi n} \quad .$$
  2. The probability that the initiator a is detected by a corrupted user (denoted by observation y_a) is
  $$C_{x_a,y_a} \quad = \quad \beta \ := \quad \frac{c(m - \varphi(n-1))}{m(m - \varphi n)} \quad ,$$
  3. The probability that a user b who is not the initiator a is detected by a corrupted user (denoted by observation y_b) is
  $$C_{x_a,y_b} \quad = \quad \gamma \ := \quad \frac{c\varphi}{m(m - \varphi n)} \qquad \text{for } a \neq b \quad .$$

Modeling the Crowds protocol

- Putting this together, we get to the following channel matrix

$$
\mathsf{C} \quad = \quad
\begin{array}{c}
 \\
x_1 \\
x_2 \\
\\
x_{n-1} \\
x_n
\end{array}
\begin{array}{c}
\begin{array}{ccccccc} y_1 & y_2 & \cdots & y_{n-1} & y_n & s \end{array} \\
\begin{bmatrix}
\beta & \gamma & \cdots & \gamma & \gamma & \alpha \\
\gamma & \beta & \cdots & \gamma & \gamma & \alpha \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
\gamma & \gamma & \cdots & \beta & \gamma & \alpha \\
\gamma & \gamma & \cdots & \gamma & \beta & \alpha
\end{bmatrix}
\end{array} ,
$$

where, as we saw,

$$
\begin{aligned}
\mathsf{C}_{x_a,s} \quad = \quad \alpha \quad &:= \quad \frac{n-\varphi n}{m-\varphi n} \quad , \\
\mathsf{C}_{x_a,y_a} \quad = \quad \beta \quad &:= \quad \frac{c(m-\varphi(n-1))}{m(m-\varphi n)} \quad , \qquad (1) \\
\mathsf{C}_{x_a,y_b} \quad = \quad \gamma \quad &:= \quad \frac{c\varphi}{m(m-\varphi n)} \qquad \text{for } a \neq b \quad .
\end{aligned}
$$

Modeling the Crowds protocol

- Now from channel matrix C, and assuming a uniform prior ϑ, we can compute the hyper-distribution $[\vartheta \triangleright \mathsf{C}]$ consisting in the following.
  - The $n+1$ posterior distributions (inners) given by

$$\delta^{y_b} = \left(\frac{\varphi}{m}, \ \ldots, \ \frac{\varphi}{m}, \ \frac{m-\varphi(n-1)}{m}, \ \frac{\varphi}{m}, \ \ldots, \ \frac{\varphi}{m}\right)$$

$$\delta^{s} = \left(\frac{1}{n}, \ \ldots, \ \frac{1}{n}\right) \quad .$$

  - The output distribution (outer) given by

$$\rho = \left({}^{1-\alpha}/_{n}, \ldots, {}^{1-\alpha}/_{n}, \alpha\right) \quad .$$

- As expected, detecting User b suggests that he is the initiator: the resulting posterior δ^{y_b} assigns higher probability to b than to the remaining users.

  E.g., in the extreme case $\varphi = 0$ users never forward, so detecting b guarantees that b is the initiator; and $\delta^{y_b} = [x_b]$ in that case.

Modeling the Crowds protocol

- Having computed the posterior distributions, we can directly express probable innocence (Def. 18.1) by requiring that

$$\delta_{x_a}^{y_b} \leq 1/2 \quad \text{for all } a \text{ and } b \ .$$

- But note that that holds just when, in particular,

$$\delta_{x_b}^{y_b} \leq 1/2 \ .$$

- Hence, the condition for probable innocence must be

$$\varphi \quad \geq \quad \frac{m}{2(n-1)} \ . \tag{2}$$

- Intuitively, probable innocence imposes a lower bound on the probability of forwarding, which in turn depends on the ratio of honest to total users.

Bayes vulnerability and Bayes leakage

Bayes vulnerability and Bayes leakage

- The Bayes vulnerability V_1 is a natural choice for measuring the anonymity guarantees of Crowds:
  - it measures the adversary's probability of successfully guessing (in one try) the initiator of the message, while
  - the Bayes leakage measures how much that probability increases due to the protocol.
- The prior case is simple:

$$V_1(\vartheta) = 1/n$$

since all (honest) users have equal probability of being the initiator.

Bayes vulnerability and Bayes leakage

- In the posterior case, we could compute $V_1[\vartheta \triangleright \mathsf{C}]$ from its definition.

  But we'll use Thm. 7.2 which says that mult. Bayes leakage for a uniform prior is the sum of the column maximums of C.

$$\mathsf{C} = \begin{array}{c} \\ x_1 \\ x_2 \\ \\ x_{n-1} \\ x_n \end{array} \begin{array}{c} \begin{array}{cccccc} y_1 & y_2 & \cdots & y_{n-1} & y_n & s \end{array} \\ \begin{bmatrix} \beta & \gamma & \cdots & \gamma & \gamma & \alpha \\ \gamma & \beta & \cdots & \gamma & \gamma & \alpha \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \gamma & \gamma & \cdots & \beta & \gamma & \alpha \\ \gamma & \gamma & \cdots & \gamma & \beta & \alpha \end{bmatrix} \end{array}$$

Since $\beta \geq \gamma$, we have

$$\mathcal{L}_1^{\times}(\vartheta, \mathsf{C}) \;=\; \mathcal{ML}_1^{\times}(\mathbb{D}, \mathsf{C}) \;=\; n\beta + \alpha \;=\; \frac{n(c+1)}{m}\,.$$

- There is a truly remarkable fact about the above equation.

  The Bayes leakage (as well as the posterior vulnerability) does not depend on the probability φ of forwarding!

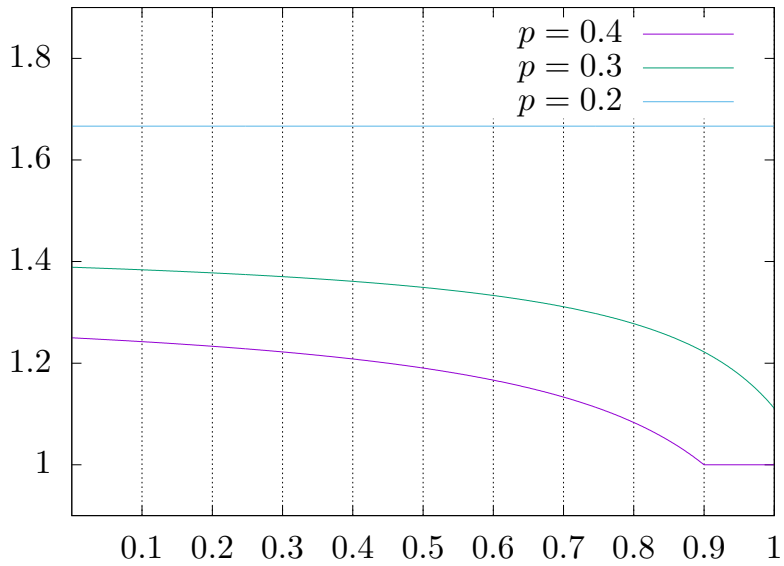
- However, the uniform prior ϑ is the only prior (apart from the degenerate cases of point priors) where this paradoxical behavior appears.

Bayes vulnerability and Bayes leakage

- The figure shows the mult. Bayes leakage of an instance of Crowds with
  - $n = 5$ honest users, and
  - $c = 1$ corrupted user.

Each value $p \in [0, 1]$ is the probability of the first (honest) user, the other four equally sharing the remaining $1-p$.

Leakage is plotted as a function of φ; each line corresponds to a prior p.

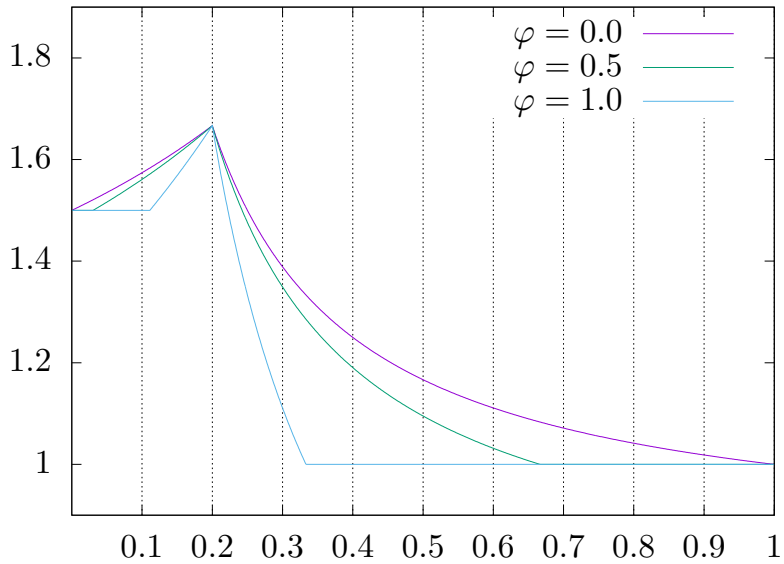


- Only the uniform case ($p = 0.2$) is independent of φ.
- In the two non-uniform cases the leakage decreases as φ increases.

  E.g., for $p = 0.4$, φ can be set so high that the adversary will always guess User 1, regardless of the output of the protocol, so there's no leakage.

Bayes vulnerability and Bayes leakage

- In this other figure for the same scenario, the leakage is plotted as a function of the prior p, for different values of φ.



- The leakage for $\varphi = 0$ is always above that of $\varphi = 0.5$.

  That, in turn, is always above the leakage for $\varphi = 1$.

- The only points where all three graphs meet are the uniform and the point priors.

- Thus the paradoxical behavior is restricted to the case of a uniform prior.

Explanation of the paradox

Explanation of the paradox

- Earlier we computed that in the original Crowds protocol, the multiplicative Bayes leakage for a uniform prior is given by

$$\mathcal{L}_1^{\times}(\boldsymbol{\vartheta}, \mathsf{C}) \quad = \quad \mathcal{ML}_1^{\times}(\mathbb{D}, \mathsf{C}) \quad = \quad n\beta + \alpha \quad = \quad \frac{n(c+1)}{m} \quad .$$

- We then remarked that the Bayes leakage (as well as the posterior vulnerability) does not depend on the probability φ of forwarding!

  How can that be possible?

  If φ is exactly what allows a detected user to plead "not guilty", how can the case $\varphi = 0$ have the same leakage as the case $\varphi = 1$?

- Here we'll explore why this apparently paradoxical behavior occurs, by studying a modified version of Crowds.

Explanation of the paradox: Modified Crowds

- Let's consider a modified version of the protocol.

  When a user forwards a message to a corrupted user, the latter can somehow know whether this forwarding happened as the first step of the route or not.

  (Of course this is a flawed protocol: it's just a tool for our analysis.)

- In the model of this modified version the set of secrets remains the same

$$\mathcal{X} \quad := \quad \{x_1, \ldots, x_n\}$$

  with x_a denoting that it was (honest) User a.

- But the set of observations becomes

$$\mathcal{Y} \quad = \quad \{(y_1, 1), (y_1, 2), \ldots, (y_n, 1), (y_n, 2), s\}, \quad \text{where}$$

  - $(y_a, 1)$ means that User a was detected in the first round of the protocol, and
  - $(y_a, 2)$ means that User a was detected in the second or a later round.

Explanation of the paradox: Modified Crowds

- Computing the modified matrix $\widehat{\mathsf{C}}$ shows the following.

1. The real initiator is detected in the 1^{st} round only if he directly forwards the message to a corrupted user, hence

$$\widehat{\mathsf{C}}_{x_a,(y_a,1)} \quad = \quad \widehat{\beta} \quad = \quad \frac{c}{m} \quad .$$

2. On the other hand, detecting any user other than the initiator in the 1^{st} round is impossible: that is because we have

$$\widehat{\mathsf{C}}_{x_a,(y_b,1)} \quad = \quad 0 \qquad \text{for } a \neq b \quad .$$

3. The probability of s remains the same, with

$$\widehat{\mathsf{C}}_{x_a,s} = \alpha \ .$$

4. Finally, after the 1^{st} round, the message is in the possession of any user with equal probability.

   Hence, due to symmetry, the probability of detecting any user in the 2^{nd} or a later round is the same, independently of who the initiator was, and

$$\widehat{\mathsf{C}}_{x_a,(y_b,2)} \quad = \quad \widehat{\gamma} \quad = \quad \frac{1-\alpha-\widehat{\beta}}{n} \quad .$$

Explanation of the paradox: Modified Crowds

- And so the modified channel matrix $\widehat{\mathsf{C}}$ is of the form

$$\widehat{\mathsf{C}} = \begin{array}{c} \\ x_1 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccccccc} (y_1,1) & \cdots & (y_n,1) & (y_1,2) & \cdots & (y_n,2) & s \end{array} \\ \begin{bmatrix} \widehat{\beta} & \cdots & 0 & \widehat{\gamma} & \cdots & \widehat{\gamma} & \alpha \\ & \ddots & & & \ddots & & \\ 0 & \cdots & \widehat{\beta} & \widehat{\gamma} & \cdots & \widehat{\gamma} & \alpha \end{bmatrix} \end{array} .$$

- We note first that the original C is a refinement of $\widehat{\mathsf{C}}$.

  We can recover the original Crowds by post-processing with a channel that forgets the extra information:

$$\widehat{\mathsf{C}} \quad \sqsubseteq \quad \mathsf{C} \quad .$$

- Hence the original version never leaks more than the modified version, which is to be expected: it can only be better, never worse.

Explanation of the paradox: Modified Crowds

- More precisely, $\mathsf{C} = \widehat{\mathsf{C}} \cdot \mathsf{R}$, or

$$
\begin{array}{c} \\ x_1 \\ \\ \\ x_n \end{array}
\begin{array}{c}
\begin{array}{cccc} y_1 & \cdots & y_n & s \end{array} \\
\begin{bmatrix} \beta & \cdots & \gamma & \alpha \\ \vdots & \ddots & \vdots & \vdots \\ \gamma & \cdots & \beta & \alpha \end{bmatrix}
\end{array}
=
$$

$$
\begin{array}{c} \\ x_1 \\ \vdots \\ x_n \end{array}
\begin{array}{c}
\begin{array}{ccccccc} (y_1,1) & \cdots & (y_n,1) & (y_1,2) & \cdots & (y_n,2) & s \end{array} \\
\begin{bmatrix} \widehat{\beta} & \cdots & 0 & \widehat{\gamma} & \cdots & \widehat{\gamma} & \alpha \\ & \ddots & & & \ddots & & \\ 0 & \cdots & \widehat{\beta} & \widehat{\gamma} & \cdots & \widehat{\gamma} & \alpha \end{bmatrix}
\end{array}
\cdot
\begin{array}{c}
\begin{array}{c} \\ (y_1,1) \\ \vdots \\ (y_n,1) \\ (y_1,2) \\ \vdots \\ (y_n,2) \\ s \end{array} \\
\begin{array}{c} y_1 \quad \cdots \quad y_n \quad s \end{array} \\
\begin{bmatrix} 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \\ 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}
\end{array}
$$

Explanation of the paradox: Modified Crowds

- A more interesting observation is that, seen as an abstract channel, Matrix $\widehat{\mathsf{C}}$ is independent of the probability φ of forwarding:

$$\widehat{\mathsf{C}}^{\varphi} \quad \sqsubseteq \quad \widehat{\mathsf{C}}^{\varphi'} \quad \sqsubseteq \quad \widehat{\mathsf{C}}^{\varphi} \qquad \text{for all } \varphi, \varphi', \tag{3}$$

  where $\widehat{\mathsf{C}}^{\varphi}$ indicates the dependence of the matrix on φ.

  - We can see that by taking a closer look at the matrix $\widehat{\mathsf{C}}$, where we notice that all columns $(y_a, 2)$ –as well as the s column– are similar.

    Indeed, they all provide no information to the adversary since they are produced with the same probability by any initiator.

  - Since $\widehat{\beta}$ is independent of φ, merging those columns together gives us a reduced matrix that is independent of φ.

  - By antisymmetry of $(\sqsubseteq)$ on abstract channels, from (3) we have the claimed independence from φ.

Explanation of the paradox: Modified Crowds

- In other words, in the original Crowds,

  The role of φ is exactly to prevent the adversary from knowing whether the detection happened in the first round or not.

- In the modified version, that information is provided explicitly.

  So φ has no effect, simply moving probability between the similar events $(y_a, 2)$ and s.

Explanation of the paradox: Vulnerability of the original protocol

- So far we know that:
  - the leakage of the modified protocol is independent of φ, and
  - it's no smaller than the leakage of the original one.

  Still, this does not explain why the Bayes leakage of the original protocol under a uniform prior is independent of φ.

- To understand that, we need to look at the formulation of posterior vulnerability as the adversary's expected gain under an optimal strategy S that maps observations to actions (Thm. 5.24):

$$V_g[\pi \triangleright \mathsf{C}] \quad = \quad \max_{\mathsf{S}} \mathbf{tr}(\mathsf{G}\lceil\pi\rfloor\mathsf{CS}) \quad . \tag{4}$$

  Here as usual **tr** is the trace operator, G is the gain function in matrix representation, the matrix $\lceil\pi\rfloor$ has π on the diagonal and 0's elsewhere, and finally S is a channel from $\mathcal{Y}$ to $\mathcal{W}$.

Explanation of the paradox: Vulnerability of the original protocol

- Let's now look at the optimal strategies S and $\widehat{\mathsf{S}}$ for Bayes vulnerability with respect to C and $\widehat{\mathsf{C}}$ in the case of a uniform prior.
  - In the original protocol S should map y_a to x_a, since detecting User a increases his chances of being the initiator.

    The s-observable provides no information, so the posterior δ^s is uniform and hence s can be mapped to any initiator.
  - In the modified protocol, an optimal $\widehat{\mathsf{S}}$ should clearly map $(y_a, 1)$ to x_a.

    The observations $(y_a, 2)$ however are similar to s: they provide no information, so the posterior remains uniform and hence it can be mapped to any initiator and not necessarily the same for each a.

    That freedom to choose any initiator is crucial to our argument: a possible choice is to map $(y_a, 2)$ to x_a.
  - That means that $(y_a, 1)$ and $(y_a, 2)$ –which are separate events in the modified protocol, but fused in the original one– will be mapped to the same secret.

Explanation of the paradox: Vulnerability of the original protocol

- Hence we have that

$$\mathsf{C}\,\mathsf{S} \quad = \quad \widehat{\mathsf{C}}\,\widehat{\mathsf{S}} \tag{5}$$

and, since the strategies are optimal, we find that for any φ, φ' we have

$$\begin{array}{lll} & V_1[\vartheta \triangleright \mathsf{C}^{\varphi}] & \\ = & V_1[\vartheta \triangleright \widehat{\mathsf{C}}^{\varphi}] & \text{“by (4),(5)”} \\ = & V_1[\vartheta \triangleright \widehat{\mathsf{C}}^{\varphi'}] & \text{“by (3)”} \\ = & V_1[\vartheta \triangleright \mathsf{C}^{\varphi'}] \quad . & \text{“by (4),(5)”} \end{array}$$

- In other words, for Bayes vulnerability and under a uniform prior, there is an optimal strategy that is independent of whether detection happens on the first or a later round, and all φ does is to confuse those two cases. Hence φ does not affect V_1.

Explanation of the paradox: Vulnerability of the original protocol

- The possibility of mapping $(y_a, 1)$ and $(y_a, 2)$ to the same secret, however, arises only under a uniform prior.
- In the case of a non-uniform prior π, say, still $(y_a, 1)$ will be mapped to x_a (because this evidence is strong enough to overcome any prior knowledge) but we no longer have that choice for $(y_a, 2)$ — the posterior is the same as π so, in order to be optimal, we need to map it to the user with the higher prior probability!
- Hence the property (5) is broken, and the leakage of the original protocol will be strictly less than that of the modified one.

  As expected, it will be dependent on φ.

Why φ matters, even for uniform priors

Why φ matters, even for uniform priors

- Comparison with the modified protocol explained why the Bayes vulnerability of the original Crowds is independent of φ under a uniform prior.
- Although the independence fails for other priors, our intuition was that, even in the uniform case, increasing φ offers better anonymity.

  The original analysis in terms of probable innocence reinforces that intuition:

  Probable innocence is satisfied only if φ is sufficiently large.

- We already discussed max-case vulnerability $V_1^{\max}$ as a possible explanation of why φ is important even for uniform priors.

  But can we have an average-case explanation of this fact?

  In other words, can we create a gain function g such that $V_g[\vartheta \triangleright C^{\varphi}]$ depends on φ?

Why φ matters, even for uniform priors

- Going back to our optimal-strategy reasoning, finding such a g is not hard.

  All we need do is construct actions $\mathcal{W}$ and gain function g in such a way that the optimal actions for $(y_a, 1)$ and $(y_a, 2)$ are necessarily different, even for a uniform prior.

- A gain function g_{lion}, very similar to g_{tiger}, behaves in just that way.

  This gain function corresponds to an adversary who is penalized for making an incorrect guess.

- As for g_{tiger}, one of the options available to such an adversary is the action $\perp$ of not making any guess at all, and her gains are given by

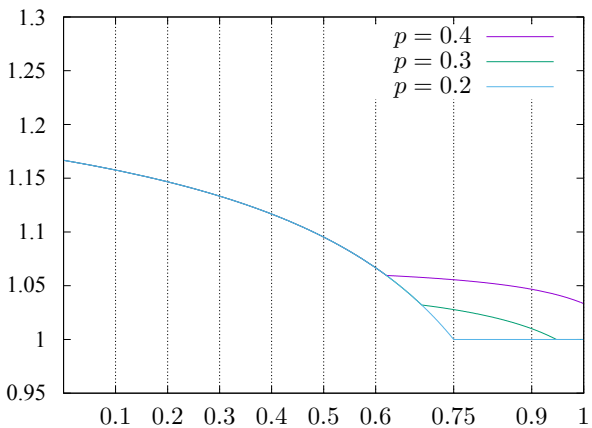
$$g_{\text{lion}}(w, x) \;:=\; \begin{cases} 1, & \text{if } w = x \\ \frac{1}{2}, & \text{if } w = \perp \\ 0, & \text{otherwise} \end{cases} .$$

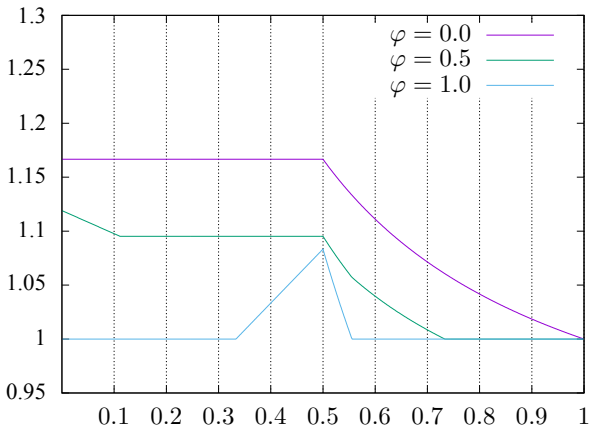
Why φ matters, even for uniform priors

- Going back to the modified Crowds, we see that
  - The "smoking gun" evidence is $(y_a, 1)$ — the adversary can be sure that a is the initiator, so she can safely guess x_a.
  - On the other hand $(y_a, 2)$ provides no evidence at all: the posterior remains uniform, and the best action remains $\perp$.

    Hence it is no longer true that the optimal strategy is independent of whether the detection happens on the first or a later round, and so the leakage no longer needs to be independent of φ.

Indeed, the dependence on φ is confirmed by the graphs for g_{lion}.





On the left-hand side we see that, even for the uniform prior given by $p = 0.2$, the leakage decreases as φ increases.

Why φ matters, even for uniform priors: Probable innocence as no lion leakage

- More interestingly, it could be argued that the g_{lion} adversary, being penalized for wrong guesses, is exactly the adversary that the original "probable innocence" property was meant to protect from.
- The idea of plausibly pleading "not guilty" assumes some hypothetical "trial" in which the user is "accused" of being the initiator.
  - A reasonable court of law clearly wants to convict the person with the highest probability of being guilty.
  - However, it doesn't have to convict anyone.

    A wrongful conviction is considered more harmful than the lack of a correct conviction, which is exactly the g_{lion} scenario.

Why φ matters, even for uniform priors: Probable innocence as no lion leakage

- Under g_{lion}:
  - The expected gain for guessing x after observing y is

    $$1 \times \delta_x^y = \delta_x^y \ ,$$

    where δ^y is the corresponding posterior.
  - On the other hand, the gain of $\perp$ is always $1/2$.

    As a consequence, as long as

    $$\delta_x^y \leq 1/2 \quad \text{for all } x, y \ ,$$

    the adversary (i.e. the court) will always choose $\perp$, which is also the best guess a priori.
- The case when the adversary never changes her guess exactly characterizes no leakage.

Why φ matters, even for uniform priors: Probable innocence as no lion leakage

- This brings us to the following result, which is a consequence of (Thm. 5.31).

Theorem (18.2)

A protocol modeled by channel C *satisfies probable innocence (Def. 18.1) just when* $\mathcal{L}^{\times}_{g_{\text{lion}}}(\vartheta, \mathsf{C}) = 1$.

- Indeed, from (2) we know that Crowds satisfies probable innocence just when $\varphi \geq {}^{m}/_{2(n-1)} = 0.75$ (for $n = 5$, $c = 1$).

  In our figures we see that this is exactly the threshold above which leakage becomes 1 in the uniform case.

Refinement: increasing φ is always safe

Refinement: increasing φ is always safe

- In the previous sections, we saw that the Bayes leakage of Crowds, for a uniform prior, is independent of the probability φ of forwarding.

  Hence, we could decrease φ (to make paths shorter), without affecting anonymity.

- This behavior, however, is very specific to the combination of g_{id} and ϑ.

  Indeed, we saw that

  - changing the prior to non-uniform, or
  - changing the gain function to g_{lion},

  both lead to a leakage that might increase when φ decreases.

  In other words, there are situations in which decreasing φ is not safe.

- This brings us to the natural and complementary question:

  "Is increasing φ always safe?"

Refinement: increasing φ is always safe

- Our intuition suggests that this is true.
  - The more we forward, the more we confuse the adversary.
  - Moreover, all the plots in the previous sections show that increasing φ decreases the corresponding leakage measures.
- But can we be sure that this always happens?

  Could it be the case that for some strange adversary, modeled by a bizarre gain function g, increasing φ causes the leakage to increase as well, leading to a less safe protocol?
- To answer this question, we employ the theory of refinement.

Refinement: increasing φ is always safe

- Similarly to the previous sections, we denote by C^{φ} the channel matrix of Crowds for a specific forwarding probability φ.
- Now consider two such values

$$\varphi \leq \varphi' \ ,$$

and let

  - α, β, γ be the corresponding elements of C^{φ}, and
  - α', β', γ' be the corresponding elements of $\mathsf{C}^{\varphi'}$.

Refinement: increasing φ is always safe

- From our previous calculation of C^{φ}, (1), we see the following.

$$\alpha \geq \alpha'$$

Intuitively, a higher probability of forwarding makes it less likely that the message will arrive at the server without detection.

- On the other hand, we have that

$$\beta \leq \beta' \quad \text{and} \quad \gamma \leq \gamma' \ .$$

Since users forward more often under φ', they are all more likely to be detected.

- Moreover, it holds that

$$\beta' - \beta \quad = \quad \gamma' - \gamma \quad = \quad \frac{\alpha - \alpha'}{n} \quad .$$

That is, the increase of probability for each observable $y_1, \ldots, y_n$ is the same in all rows of the matrix.

Refinement: increasing φ is always safe

- This means that we can convert C^{φ} to $C^{\varphi'}$ by reducing the probability of s and equally redistributing it to $y_1, \ldots, y_n$.

  That can be achieved by post-processing C^{φ} with the following channel

$$R := \begin{array}{c} \\ y_1 \\ \vdots \\ y_n \\ s \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & \cdots & y_n & s \end{array} \\ \begin{bmatrix} 1 & \cdots & 0 & 0 \\ & \ddots & & \\ 0 & \cdots & 1 & 0 \\ \kappa & \cdots & \kappa & \lambda \end{bmatrix} \end{array} , \quad \text{where}$$

$$\kappa := \frac{\alpha - \alpha'}{n\alpha} , \quad \lambda := \frac{\alpha'}{\alpha} .$$

It is easy to verify that R is indeed a valid channel matrix and that

$$C^{\varphi} R = C^{\varphi'} .$$

Refinement: increasing φ is always safe

- That means that increasing the probability φ produces a refinement of the original protocol.

  It is indeed a safe operation in the sense that for any prior and gain function it can only decrease the protocol's leakage.

- The reason the above fails when $\varphi > \varphi'$ is that in that case the matrix R is not a valid channel matrix, because $\lambda > 1$.

  In fact, it can be shown that no channel R such that $C^{\varphi} R = C^{\varphi'}$ exists.

- Hence, reducing φ is not always a safe operation, although it might be safe in some special cases –as in the case of Bayes leakage under a uniform prior– but there will always be some adversary for which leakage increases.

Refinement: increasing φ is always safe

- The above discussion is summarized in the following result.

Theorem (18.3)

For Crowds-protocol channels C^{φ} and $\mathsf{C}^{\varphi'}$ we have $\mathsf{C}^{\varphi} \sqsubseteq \mathsf{C}^{\varphi'}$ iff $\varphi \leq \varphi'$.

Proof. The "if" part comes from the construction $\mathsf{C}^{\varphi}\mathsf{R} = \mathsf{C}^{\varphi'}$ above.

For the "only if", assume that $\varphi > \varphi'$. We could show that no such channel R exists, but it is in fact easier to use Thm. 9.11 by constructing a counterexample gain function under which $\mathsf{C}^{\varphi'}$ leaks strictly more than C^{φ}.

Let $t = V_1^{\max}[\vartheta \triangleright \mathsf{C}^{\varphi}]$. From and $\varphi > \varphi'$, we know that $t < V_1^{\max}[\vartheta \triangleright \mathsf{C}^{\varphi'}]$.

Then define g_t as g_{id} with an extra action $\perp$ such that $g_t(\perp, x) = t$ for all x in $\mathcal{X}$.

From Thm. 5.31 we get that $V_{g_t}[\pi \triangleright \mathsf{C}^{\varphi}] = t < V_{g_t}[\pi \triangleright \mathsf{C}^{\varphi'}]$.

□

Multiple paths

Multiple paths

- So far we have studied the anonymity of Crowds when a single path is established from the initiator to the server.
- But what happens if a user establishes multiple paths to the same server?
  - On the one hand, using multiple paths could be desirable, for instance for load-balancing purposes.
  - But more importantly, establishing a new path could be forced by an adversary controlling either the server or intermediate users, by simply ceasing to respond, thus rendering the original path useless.
- Note that we assume that the adversary can always link distinct paths as having originated from the same original user (although she does not know which one).
  - That could be done by inspecting either the content of the request (which might be in plaintext), but also by using metadata such as its size or timing.

Multiple paths

- The anonymity of the protocol crucially depends on how the new path is created.

  We consider two cases:

  1. The new path is recreated from scratch by the initiator, by restarting the protocol (either voluntarily, or after detecting a failure).
  2. The old path is kept until the last working node, which detects the failure and tries to "repair" the path by removing the faulty node from its list, randomly selecting a new node and forwarding the request to him.

- We'll analyze the two cases in detail.

  Each requires different tools from the QIF theory.

Multiple paths: Paths created by the initiator

- When a path is recreated by the initiator, the whole protocol is executed from scratch, producing a new run of the original channel under the same secret.
- If k paths are created then the protocol can be modeled as $\mathsf{C}^{(k)}$, which are the k independent runs of C.

  That's the parallel composition of C with itself k times,

$$\mathsf{C}^{(k)} = \mathsf{C} \parallel \ldots \parallel \mathsf{C} \ .$$

- That allows us to apply the following result bounding the multiplicative Bayes-capacity of the parallel composition of two arbitrary channels. (Its proof is Exercise 8.11).

Theorem (18.4)

Given compatible channels C_1 *and* C_2*, we have*

$$\mathcal{ML}_1^{\times}(\mathbb{D},\, \mathsf{C}_1 \parallel \mathsf{C}_2) \quad \leq \quad \mathcal{ML}_1^{\times}(\mathbb{D}, \mathsf{C}_1) \times \mathcal{ML}_1^{\times}(\mathbb{D}, \mathsf{C}_2) \quad .$$

Multiple paths: Paths created by the initiator

- Applying that to Crowds we get that

$$\mathcal{ML}_1^{\times}(\mathbb{D}, \mathsf{C}^{(k)}) \quad \leq \quad \mathcal{ML}_1^{\times}(\mathbb{D}, \mathsf{C})^k \quad = \quad \left(\frac{n(c+1)}{m}\right)^k \quad .$$

- Unless $c = 0$ (in which case the protocol has no leakage at all), that bound grows with increasing k.
- For small values of k, that (together with the "Miracle" Thm. 7.5) provides an easy way of bounding the leakage of the protocol.

  For large values of k, however, the bound becomes too loose; indeed, it can be shown that $\mathcal{ML}_1^{\times}(\mathbb{D}, \mathsf{C}^{(k)})$ converges asymptotically to the number of distinct rows of C.

  For Crowds that is n, meaning that in the long run the system completely exposes the identity of the initiator.

Multiple paths: Paths created by the initiator

- If we are interested in a precise analysis instead of bounds, we can compute directly the desired leakage measure for $\mathsf{C}^{(k)}$.

  For instance, the max-case Bayes vulnerability will be given by the observation k times of the same User b, that is

$$V_1^{\max}[\vartheta \triangleright \mathsf{C}^{(k)}] \quad = \quad V_1(\delta^{y_b,\ldots,b}) \quad = \quad \frac{1}{1+(\gamma/\beta)^k} \quad .$$

- Again, we see that repeated executions increase leakage; as k grows the vulnerability converges to 1.

  If we are interested in probable innocence we can compute the largest k for which this vulnerability remains below $1/2$.

- Based on the analysis above, we conclude that paths should remain static and new ones should be created by the initiator as rarely as possible.

  In their original design, Reiter and Rubin propose creating new paths only when new users join the Crowds, which should happen in infrequent scheduled events (the purpose being to protect the anonymity of the new users).

Multiple paths: Paths repaired by the last working node

- An adversary controlling any node in the path can force it to be recreated by simply stopping the forwarding of any traffic.
- However, recreating the path from scratch causes severe information leakage, as we saw in the previous section.
- To cope with that, broken paths in Crowds are not recreated by the initiator, but are instead repaired by the last working node, by choosing a new node and forwarding the request to him.
- New forwarding requests can of course reach corrupted users, so that poses the natural question: does this operation increase information leakage?

Multiple paths: Paths repaired by the last working node

- In their original paper, Reiter and Rubin state:

  "Since the [corrupted users] cannot distinguish whether that predecessor is the initiator or not, the random choices made by that predecessor yield no additional information to the [adversary]."

- Although the intuition is clear, the argument remains quite informal.

  Having seen that repeated executions highly affect leakage, we would like to have a robust formal argument that repairing paths in this way is safe under any circumstances.

Multiple paths: Paths repaired by the last working node

- Let $\widehat{\mathsf{C}}$ be the channel modeling the protocol when the adversary forces the path to be recreated once.

  Its secrets $\mathcal{X}$ are the same as for C, but its observations are now $\mathcal{Y}\times\mathcal{Y}$; the observation (y_a, y_b) means that User a was detected when the path was first created, while User b was detected when the path was repaired.

  The crucial observation is that we can obtain $\widehat{\mathsf{C}}$ by post-processing C with a channel R from $\mathcal{Y}$ to $\mathcal{Y}\times\mathcal{Y}$ that, given an observation y_a of C, restarts the protocol (after removing the faulty node) from User a, obtains a new observation y, and then outputs both y_a, y_b.

Multiple paths: Paths repaired by the last working node

- More precisely, let C' be the channel modeling Crowds with $c-1$ corrupted users (since the non-responding corrupted node was removed), and let R be defined as

$$\begin{aligned} R_{y_a,(y_a,y_b)} &= C'_{x_a,y_b} \qquad \text{for } x_a \text{ in } \mathcal{X} \text{ and } y_b \text{ in } \mathcal{Y}, \\ R_{s,(s,s)} &= 1 \quad , \end{aligned}$$

  and 0 elsewhere.

  It is then easy to see that $\widehat{C} = CR$.

- Moreover, $\widehat{C}$ can be trivially post-processed back to C (by ignoring the second observation), hence $C \sqsubseteq \widehat{C} \sqsubseteq C$.

  In other words C and $\widehat{C}$ are different representations of the same abstract channel, which means that their leakage is exactly the same.

  Using the same argument we can conclude that any number of repaired paths causes no further leakage, in sharp contrast with the analysis of the previous section.

Multiple paths: Multiple detections and deviating from the protocol

- In our analysis of Crowds we assumed that, in each execution, at most one user is detected by a corrupted node.

  This is equivalent to assuming that corrupted nodes always deliver the message to the server, as well as communicating with the adversary.

- However, if instead a corrupted node forwards the message to another user, it is possible to detect further users in the extended path if they too choose to forward to a corrupted node.

  Do those extra observations increase information leakage?

- Moreover, a corrupted node might not be obliged to execute the protocol faithfully.

  It could for instance forward the message to a specific user of its choice.

  Could such deviations help the adversary in any way?

Multiple paths: Multiple detections and deviating from the protocol

- We can answer those questions in a way similar to the analysis of the previous section.
  - If corrupted nodes cannot affect the actions of honest nodes, then a channel modeling deviations from the protocol can be obtained by post-processing the original C with a channel R, just as above.
  - And that will always lead to a refinement of C that (therefore) cannot leak more information than C itself under any circumstances.