Calculating Probabilistic Anonymity from Sampled Data

Konstantinos Chatzikokolakis
Dept. of Computer Science
Technische Universiteit Eindhoven
k.chatzikokolakis@tue.nl

Tom Chothia
School of Computer Science
University of Birmingham
T.Chothia@cs.bham.ac.uk

Apratim Guha
School of Mathematics
University of Birmingham
guhaa@for.mat.bham.ac.uk

Abstract

This paper addresses the problem of calculating the anonymity of a system statistically from a number of trial runs. We show that measures of anonymity based on capacity can be estimated, by showing that the Blahut-Arimoto algorithm converges for sampled data. We obtain bounds on the error of the estimated value by calculating the distribution of mutual information when one distribution is known and one unknown. This leads to finding the variance of the estimation of anonymity in terms of the numbers of samples, inputs and possible observations, which in turn tells us what kinds of systems can and cannot be accurately analysed using a statistical approach. We demonstrate our method on an implementation of the Dining Cryptographers protocol and on a Mixminion anonymous remailer node.

1. Introduction

Anonymous communication systems are becoming increasingly popular and important as Internet users become aware of the threats to their personal information. However, the anonymity offered by these systems is often poorly understood. This lack of understanding is especially true when it comes to *probabilistic* behaviour. One system might hide users completely and another system might just introduce a small amount of doubt as to the users' true identity, yet both systems could be said to offer "anonymity". Understanding this distinction can become even more complicated when the level of anonymity a system provides is affected by how the system is used.

Information theory provides powerful techniques to measure the relation between different probability distributions (see e.g. [11]) and so has proved useful for defining anonymity ([26], [14], [28], [23], [7]). Following the work of Moskowitz et al. [23] and Chatzikokolakis et. al [7], we base our definition of anonymity on the information theoretic notion of capacity. This approach has the advantage of accounting for any previous knowledge the attacker might have about the system. In particular, we do not need to assume that the users of a system are all equally likely to be guilty or that they use the system in a uniform way, which is unlikely to be the case in any real system.

Previous work using capacity and mutual information to measure anonymity has assumed that the exact behaviour of the system, that is the probability of each observation under any user, is known. Typically, one has to construct a model of the system and use a model checker to compute the actual probabilities. In this paper we address the problem of how measures of anonymity can be calculated from trial runs of the system alone. There are three main reasons to base our method on sampled data, rather than say the output of a formal model.

First, information security systems tend to be complex and many attacks are based on subtle aspects like time or quality of service, therefore creating an accurate model may be very difficult. Second, when we do have a model, as the number of users increases the internal state space quickly becomes too big to be handled by model checking tools (a problem even harder for probabilistic model checking). Third, it is often the case that an attack exploits implementation faults and not a problem in the protocol itself; an example of such a scenario is discussed in Section 9, where a flawed implementation of the Dining Cryptographers protocol is analysed. Clearly, such attacks can be only discovered by analysing the implementation itself.

The user of our method must define the inputs of the system, which correspond to the events that we wish to keep anonymous, and the possible observations an attacker might make, which corresponds to defining the appropriate attacker model. The system under test is then run a number of times until an estimated probability transition matrix can be built up. We apply the Blahut-Arimoto algorithm ([1], [5]) to this matrix in order to estimate the capacity and hence the anonymity of the system. As the Blahut-Arimoto algorithm finds the input distribution that maximises the information leakage we may generate our sample data using a uniform prior distribution and then let the algorithm find the worst possible usage.

We prove that our estimate converges to the true value of capacity. However, this doesn't answer the question of how close the result of a single test is to the true value. We find a lower and upper bound for the true value of capacity in terms of our estimate, based on the largest likely error of any entry in the matrix. To provide a much more accurate lower bound we find the distribution that our estimate comes from. This turns out to be a χ^2 distribution in the case that the capacity is zero and a normal distribution if the capacity is non-zero. In the latter case, the best estimate is the mean of the distribution plus a small correction. In finding this result, we solve the more general problem of finding the distribution of mutual information between two random variables, when the probability distribution of the one is known and the other is not. This result also makes it possible to estimate the mutual information of a system for uniform usage, or any other given prior.

The variance of the estimate is dominated by the number of inputs times the number of outputs, divided by the number of samples. Therefore, a statistical estimate will be accurate if there are significantly more samples than the product of the number of inputs and all observable outputs. The ability to generate this many samples, in a reasonable amount of time, acts as a guide to which systems can and cannot be analysed statistically. Note that this can still be much more efficient than model-checking. Often, complex systems have a great number of "internal" states, but still generate few observations. In this case, generating samples is easier than constructing the whole state space of the system. Moreover, even if the number of observations is too big, we could concentrate on some of them and still be able to make a partial, yet useful analysis of the system.

We have implemented a tool to calculate the estimate of anonymity directly from a list of observations of a system. This tool calculates the estimated capacity of a system, tests it for consistency with zero and if the evidence suggests that it is non-zero then it calculates bounds on the true value.

We illustrate our method by applying it to an implementation of the Dining Cryptographers protocol, as well as to a Mixminion mix node. Dining Cryptographers is often proved correct in order to demonstrate formal anonymity frameworks. However, as we show here, an implementation could lose anonymity due to factors such as scheduling or extra work being required of the payer. Mixminion is a state-of-the-art mix network. As a simple example of the application of our method to such a system we test if anything about the order in which messages entered the mix can be deduced from the observation of packets leaving the mix. We ran our own test node and sent messages across the Internet to other mix nodes. We used a packet sniffer to observe all data leaving our node. Our results are consistent with the Mixminion node having perfect anonymity with respect to the order in which single messages arrive and are forwarded.

The contributions of this paper are:

- Showing that anonymity, as defined by capacity, can be calculated from sampled data, by showing that the Blahut-Arimoto algorithm converges for sampled data.
- Proving bounds on the error of the estimate, which decrease as the sample size increases.
- Finding a bound on the variance of our estimate, and so establishing what types of systems can and cannot be meaningfully analysed using a statistical approach.
- Illustrating our method and providing evidence that our distribution calculations are correct by analysing an implementation of the Dining Cryptographers protocol and a Mixminion remailer node.

The next section discusses background and related work. We describe how we can calculate an estimate of anonymity from sampled data in Section 4. We find bounds on our estimate of anonymity in Section 5 and we establish the distribution that our estimate is drawn from in Section 6. In Section 7 we briefly discuss tool support for our method and we demonstrate this with an implementation of the Dining Cryptographers protocol in sections 8 and 9. We show that our method can be applied to real software in Section 10 by analysing a Mix node and we conclude and discuss further work in Section 11. Full proofs are given in the appendix.

2. Information-theoretic measures of anonymity

Information theory reasons about the uncertainty of a random variable and the information we can obtain for a variable X by observing a variable Y. Let X, Y be random variables and $\mathcal{X}, \mathcal{Y}$ their set of values. The *entropy* of X is defined as:

$$H(X) = -\sum_{x\in\mathcal{X}} p(x)\log p(x)$$

and measures the uncertainty about its outcome. It takes the minimum value 0 when X is constant (gives a certain value with probability 1) and the maximum value $\log|\mathcal{X}|$ when its distribution is uniform. The *conditional entropy* is defined as:

$$H(X|Y) = -\sum_{y\in\mathcal{Y}} p(y)\sum_{x\in\mathcal{X}} p(x|y)\log p(x|y)$$

and measures the uncertainty about X that "remains" when we know Y. It takes its minimum value 0 when Y completely determines the value of X (eg, when $X = Y$) and its maximum value $H(X)$ when X, Y are independent. Then the *mutual information* between X, Y, defined as

$$I(X;Y) = H(X) - H(X|Y)$$

measures the information that we learn about X if we observe Y. It is symmetric ($I(X;Y) = I(Y;X)$) and ranges between 0 (when X, Y are independent) and $H(X)$ (when X, Y are totally dependent). Finally, the *Kullback-Leibler* distance between distributions p, q is defined as

$$D(p \parallel q) = \sum_x p(x)\log\frac{p(x)}{q(x)}.$$

A *channel* consists of an input alphabet $\mathcal{X}$, an output alphabet $\mathcal{Y}$ and a probability matrix W where $W_{x,y} = p(y|x)$ gives the probability of output y when x is the input. Given a channel and an input distribution on $\mathcal{X}$, we can define two random variables X, Y representing the input and output of the channel, and with a slight abuse of notation we write $I(X, W)$ for $I(X;Y)$. The *capacity* of the channel is defined as the mutual information between the input and the output, maximised over all input distributions:

$$C = \max_{p(x)} I(X, W)$$

Anonymous communication channels are inherently probabilistic. In such systems, we have a set $\mathcal{A}$ of *anonymous events* that we wish to keep hidden, for example the identities of the users or the link between senders and receivers. Moreover, we have a set $\mathcal{O}$ of *observable events* which model what the attacker can observe about the protocol. We assume that on each execution, exactly one $a \in \mathcal{A}$ and $o \in \mathcal{O}$ will happen and that the output of the protocol is chosen probabilistically. Then we can define the random variables A, O representing the anonymous and observable events, and use the aforementioned notion to measure the anonymity guarantees of the protocol.

The choice of $\mathcal{A}, \mathcal{O}$ should be made by the user of our method. $\mathcal{A}$ is determined by the anonymity requirements of the protocol, since it contains the events that the protocol aims at keeping anonymous. On the other hand, the choice of $\mathcal{O}$ is more involved and corresponds to the selection of an attacker model. As usual, the analysis of a system can be done at a finer or coarser level. A finer attacker model will lead to a larger set $\mathcal{O}$, allowing us to discover more attacks at the cost of a greater number of required samples. Note also that our method is mostly suitable for analysing passive attackers, who only observe the output of a protocol without interfering with it. However, the analysis of an active attacker is also possible if the attack can be simulated and samples of the system under the attack can be produced.

The first information-theoretic anonymity metric was given by Serjantov and Danezis [26] and independently by Diaz et al [14]. They define the degree of anonymity as the entropy of the distribution that the attacker assigns to the users after observing the protocol. This corresponds to $H(A|O)$ (although in the above papers the model is slightly different and there is no explicit mention of observable events). A higher entropy means that the attacker is uncertain about the anonymous events, so the system offers anonymity.

One disadvantage of the above definition is that it depends on the distribution of A (the probability of each user performing the action of interest) or in other words the prior knowledge that the attacker has about the users. For example, if one user sends messages much more often than other users, then the attacker will naturally assign a higher probability to him, but this is not a problem of the protocol. Consequently, we usually need to assume a uniform distribution of anonymous events, which might be problematic in cases where the maximum leakage happens for a non-uniform input distribution. Zhu and Bettati [28] propose the use of mutual information $I(A;O)$ as a measure of anonymity. This definition measures how much information about the anonymous events is revealed by the attacker's observation, and is not limited to a uniform prior distribution. Still, it requires that a certain prior is fixed.

The next natural step is to maximise the mutual information over all input distributions, which brings us to the notion of capacity. Moskowitz et al [23] are the first to suggest capacity as a measure of anonymity. They note that covert channels of non-trivial throughput can be created as a result of non-perfect anonymity. Then they suggest that the capacity of those channels, called *quasi-anonymous* channels, can be used as the degree of anonymity of the protocol. Chatzikokolakis et al [7] define channels where the inputs and outputs consist of the anonymous and observable events respectively. The capacity of such channels is a natural

	DAAA	ADAA	AADA	AAAD	DDDA	DDAD	DADD	ADDD
user1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
user2	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
user3	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
user4	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125

(a) Fair coins

	DAAA	ADAA	AADA	AAAD	DDDA	DDAD	DADD	ADDD
user1	0.320	0.117	0.070	0.117	0.117	0.070	0.117	0.070
user2	0.117	0.320	0.117	0.070	0.070	0.117	0.070	0.117
user3	0.070	0.117	0.320	0.117	0.117	0.070	0.117	0.070
user4	0.117	0.070	0.117	0.320	0.070	0.117	0.070	0.117

(b) Equally unfair coins (to 3 decimal places)

	DAAA	ADAA	AADA	AAAD	DDDA	DDAD	DADD	ADDD
user1	0.17	0.08	0.17	0.17	0.08	0.08	0.17	0.08
user2	0.08	0.17	0.08	0.08	0.17	0.17	0.08	0.17
user3	0.17	0.08	0.17	0.17	0.08	0.08	0.17	0.08
user4	0.17	0.08	0.17	0.17	0.08	0.08	0.17	0.08

(c) 2 unfair coins

Figure 1. Probability transition matrices for the Dining Cryptographers protocol

measure of anonymity: it measures how much information about the anonymous events is revealed by the observation, in the case of the worst prior distribution. As this metric for anonymity avoids the problems with having to assume knowledge of the input distribution and can be applied to a wide range of situations we use it as the basis of our analysis.

A number of authors have used capacity or conditional entropy as a measure of information flow ([21], [20], [10], [18], [2]). Typically, inputs to the channel are the secret, high values and the outputs are the low, public values; the capacity then equals the amount of information leaked. Our approach may be also useful in this setting, for estimating information flow from sampled data.

The Blahut-Arimoto algorithm. In the general case, there is no analytical formula for capacity. So we use the iterative Blahut-Arimoto algorithm [1], [5] that can compute the capacity of an arbitrary channel to within a given precision. To explain this algorithm we first observe that mutual information can be written in terms of relative entropy:

$$\begin{aligned} I(Q,W) &= H(Q) - H(Q|Y) \\ &= \textstyle\sum_x \sum_y Q(x)W(y|x) \log\left(\frac{W(y|x)}{\sum_{x'} Q(x')W(y|x')}\right) \\ &= \textstyle\sum_x Q(x) D(W(\cdot|x) \parallel \sum_{x'} Q(x')W(\cdot|x')) \end{aligned}$$

We write $D_x(W \parallel QW)$ as short hand for $D(W(\cdot|x) \parallel \sum_{x'} Q(x')W(\cdot|x'))$. This leads to an upper bound for capacity; by observing that, for any set of numbers $\{n_1, \ldots, n_m\}$ and any probability distribution $\{p_1, \ldots, p_m\}$ it holds that $\sum_i p_i.n_i \le \max_i n_i$, we find that, for all probability distributions Q:

$$\sum_x Q(x) D_x(W \,\|\, QW) \le C(W) \le \max_x D_x(W \,\|\, QW) \tag{1}$$

It can be shown [5] that these inequalities become equalities when Q is the input distribution that achieves capacity.

The term $D_x(W \parallel QW)$ can be thought of as a measure of the effect that choosing the input x has on the output. Blahut and Arimoto showed that the maximising input distribution could be found by repeatedly increasing this measure. Given a channel W, the algorithm starts from an initial input distribution Q^0 (we start from a uniform one, if no better one is known) and in each step k we obtain a new distribution Q^{k+1} by updating the current Q^k for each input x as follows:

$$Q^{k+1}(x) = Q^k(x) \frac{\exp(D_x(W \parallel Q^k W))}{\sum_{x'} Q^k(x') \exp(D_{x'}(W \parallel Q^k W))}$$

The algorithm is guaranteed to converge to the capacity achieving distribution Q. Furthermore, (1) can be used as a stopping criterion as for any $\epsilon \ge 0$, terminating the iterations when $\max_x D_x(W \parallel Q^k W) - I(Q^k, W) \le \epsilon$ ensures that the estimate is within ϵ of the true capacity, with equality when the capacity has been found (i.e., $Q^k = Q$).

Matz and Duhamel [19] propose an accelerated algorithm with the update:

$$Q(x)^{k+1} = Q(x)^k \frac{\exp(\mu_k.D_x(W \parallel Q^kW))}{\sum_{x'} Q(x')\exp(\mu_k.D_x(W \parallel Q^kW)}$$

where $\mu_k = D(Q^kW \parallel Q^{k-1}W)/D(Q^k \parallel Q^{k-1})$. They demonstrate super-linear convergence for this algorithm, and prove at worst linear in the general case.

The basic Blahut-Arimoto algorithm has been extended to calculate the capacity of channels with side information [15] and channels with memory [27], amongst others. The Blahut-Arimoto algorithm is not the only way to find capacity (although it does seem to be the fastest and most general). Other options include direct calculation for special cases (e.g. [7]), a gradient search using the Frank-Wolfe algorithm or the Kuhn-Tucker Theorem/Lagrange Multipliers. Pasquale and Chen [17] have used Lagrange multipliers to show how capacity can be found under certain constraints, such as one particular user being twice as likely to send as another.

An example: the DC protocol. The Dining Cryptographers protocol [9] is a typical example of an anonymous protocol. In this protocol, one cryptographer is the payer and wants to communicate this fact to the others, while staying anonymous. One cryptographer might be much more likely to pay for the dinner than the others, so asking how likely each cryptographer was to be the payer after observing a run of the protocol does not tell us much about the protocol itself. So, to analyse the system independently of how it is used, we base our analysis on the conditional probabilities of each possible observation given each possible payer.

The conditional probabilities for three different versions of the Dining Cryptographers protocol are shown in Figure 1 (to three decimal places). Here there are 4 users and 6 possible observable actions. Each element of the matrix gives the probability of each observable action in the case that the corresponding user is the payer. Figure 1(a) shows the results of the protocol based on fair coins. All the outcomes are equally likely and so nothing can be learnt from the observations: anonymity is ensured. Figure 1(b) shows the results of the protocol if the cryptographers use coins that come up heads three quarters of the time. As the coins are now more likely to agree, any cryptographer that says disagree is suspect. Figure 1(c) shows the result of running the protocol with two honest coins and two coins that come up heads four out of five times. For full details of the Dining Cryptographers protocol we refer the reader to Chaum's original paper [9].

Clearly, the system is most anonymous with fair coins, no observation is more likely than any other under any circumstances, but the other two instances of the protocol are harder to analyse. Capacity can be used to find the maximum amount of data that can be sent through a channel, i.e., what is the maximum possible information that you can learn about the inputs from looking at the outputs. If we look at the decision of who is guilty as the inputs of a channel and the observable actions of a system as the outputs, then information theory will give us an absolute upper bound on what an attacker could learn about the anonymous events, under the assumption that the set of observations contains everything the attacker can observe.

From the probability matrix of the channel we can calculate its capacity by using the Blahut-Arimoto algorithm. Applying it to the matrices given in Figure 1, we find that, to four decimal places, the first system in Figure 1(a) has a capacity of zero, indicating perfect anonymity. The system in Figure 1(b) has a capacity of 0.1705 whereas the system in Figure 1(c) has a capacity of 0.0956, which is achieved when user2 is 50% more likely to be the sender then the other two. Hence, the protocol is better with two very biased coins than four somewhat biased coins. If we had analysed this system assuming that each principal was equally likely to be the payer we would have underestimated the worst-case possible information leakage of the system in Figure 1(c).

3. System model and assumptions

A system in our framework consists of a set of secret inputs (or anonymous events) $\mathcal{X}$, a set of observable output actions $\mathcal{Y}$ and a probability transition function W that describes the behaviour of the system. We require that, given one particular secret input, the system must act as a probability distribution on the outputs. This means that if we run the system W with the input x then there must be a fixed probability of seeing each observable output ($W(y|x)$ equals the probability of seeing y given x).

In statistical terms, given a configuration of the system x, the trial runs of the system must be independent and identically distributed. Factors other than the anonymous event x, that and not accounted for by the probabilities of the outputs, must not have a statistically significant effect on the observed actions of the system.

We consider a passive attacker that observes the outputs of the system and may try to make deductions

W	:	the true probability transition matrix for the system
$\hat{W}_n$	:	an estimated probability transition matrix for the system from n samples
Q	:	the input distribution that maximises mutual information for W
$Q(\hat{W}_n)$	:	the input distribution that maximises mutual information for a $\hat{W}_n$
$C(W) = I(Q, W)$	:	the true capacity of W
$C(\hat{W}_n) = I(Q(\hat{W}_n), \hat{W}_n)$	:	the true capacity of the matrix found by sampling
$\hat{Q}_m(\hat{W}_n)$	:	the result of running the Blahut-Arimoto algorithm on $\hat{W}_n$ for m iterations
$\hat{C}(\hat{W}_n) = I(\hat{Q}_m(\hat{W}_n), \hat{W}_n)$	:	our estimate of the capacity of W

Figure 2. Key values for estimating capacity

from these outputs but does not interact with the system directly. Capacity measures the most information that can be sent over a channel, no matter how it is used, so we do not require anything about the distribution of anonymity events. How the channel is used does not have to be independent of previous usages. As long as the observer does not have prior knowledge of the sequence of anonymity events, they will not be able to work out more information than the capacity [11].

Given these assumptions, our analysis estimates the information leakage as the information-theoretical capacity of W. This is the maximum amount of information, in bits, that can be passed over W when it is regarded as a communication channel. In terms of anonymity, it is the maximum number of bits that the attacker can learn about which anonymity event took place, on average, from observing the system. An information leakage of $log_2(\#\mathcal{X})$ means that the system offers no anonymity, whereas an information leakage of 0 means that the system is perfectly anonymous. A capacity in between these values indicates a partial loss of information. As with any single value measure of anonymity we do not distinguish between a small chance of a total loss of anonymity and a high probability of a partial loss, rather our figure represents the average case for the average user, as is common in information theoretical analysis. We also note that a statistical approach is ill suited to any measure that rates a tiny probability of a total loss of information as much worse than no loss of information because such a measure would not be continuous as the probability tended to zero and so would not allow for a accurate confidence intervals to be found.

Our analysis method makes no assumptions about the distribution on anonymous events and assesses the whole system; this means that our results are valid no matter how the system is used but they cannot say anything about particular observed runs of the system. To do so would require one to make assumptions about the prior distribution, for instance as part of a Bayesian analysis [3]. Such an analysis (e.g. [8]) gives the probability of identifying the culprit from given observations, but would not be valid if the assumptions are wrong or the users' behaviour changes.

4. Estimating anonymity

In this paper we focus on capacity as our measure of anonymity, we now describe how it can be calculated. There are two main obstacles to finding the capacity of a real system: first, we must find a probability transition matrix that reflects the system under test and gives the conditional probabilities of any observable action (the outputs) given a particular usage of the system (the inputs). Second, we must calculate capacity from this estimated matrix.

To find the probability transition matrix we start by defining the inputs (the events that we wish to keep anonymous) and the outputs (the actions observable to an attacker). As explained in Section 2, the latter corresponds to defining an attacker model. Some level of abstraction must be used; this choice should be made by the user of our method, depending on the needs of the analysis. Our method requires many more samples than the number of observations so the more fine grained the attacker's observations are, the more samples we require; we quantify this in Section 6 where we calculate the variance of our results in terms of the number of inputs, outputs and samples. Defining the input and output of the channel is a challenging task and it should be approached with some care, because it greatly influences the result of the analysis. Our method deals with the next step, which is computing anonymity in an automated way after the channel has been fixed.

Once the inputs and outputs are identified, we may run trials of the system for each of the inputs and record the observable outcomes. We use these observations to construct an estimated matrix. Note that the approximate matrix can be generated using any probability distribution on the inputs, that is without making any assumption about how the system is used. Calculating the capacity then finds the input distribution that leaks the most information. Therefore, we can collect our data for any usage of the system and then calculate the worst-case scenario.

There are two sources of error in the method we propose. The first comes from estimating the probability transition matrix for the system and the second from the approximation of capacity based on this matrix. Running a numerical approximation on inaccurate data does not necessarily lead to meaningful results, but we prove below that running the Blahut-Arimoto algorithm on an approximate matrix does return a result that tends to the true capacity as the sample size and the number of iterations increases.

The values and distributions used in our results are summarised in Figure 2. Our analysis of a system is based on the probability transition matrix W that gives conditional probabilities of each input given each output, $W(o|a) = p(o|a)$, i.e., the probability of the attacker seeing observation o given that the system is started in configuration a. We will estimate W by running the system n times with a uniform random input each time. This leads to an estimate $\hat{W}_n$, which is a matrix drawn from a normal distribution with mean W and a variance that decreases as n increases.

Next, we have the input distribution that maximises the mutual information for W, which we label Q. The true capacity of the system C is given by the mutual information for input Q, denoted by $C = I(Q, W)$. Only in certain cases can we find Q exactly, so we estimate Q using the Blahut-Arimoto algorithm for m iterations; we write $\hat{Q}_m(W)$ for this distribution. We may also apply the Blahut-Arimoto algorithm to our estimated matrix to get $\hat{Q}_m(\hat{W}_n)$ which converges to the input distribution that maximises mutual information for the estimated matrix $\hat{W}_n$. This leads to our estimate of capacity for the system: $\hat{C}(\hat{W}_n) = I(\hat{Q}_m(\hat{W}_n), \hat{W}_n)$.

The following theorem tells us that this estimate is good, i.e., with enough samples and iterations of the Blahut-Arimoto algorithm our estimate of capacity almost surely converges to the true value:

Theorem 4.1: For any probability $p_e > 0$ and any real number $e > 0$ there exists integers n', m' such that for all $n > n'$ and $m > m'$ and for an estimated probability transition matrix found using n samples $\hat{W}_n$ it holds that

$$p(|I(\hat{Q}_m(\hat{W}_n), \hat{W}_n) - I(Q, W)| > e) < p_e$$

Proof Sketch: Our proof is by contradiction. We assume that $\hat{C}$ does not almost surely converge to C. Mutual information is continuous and finite for a fixed number of inputs therefore our assumptions imply that there must also be a difference between $I(Q(\hat{W}_n), W)$ and $I(Q, W)$ or between $I(Q, \hat{W}_n)$ and $I(Q(\hat{W}_n), \hat{W}_n)$, however if these differences exist then either $Q(\hat{W}_n)$ does not maximise mutual information for $\hat{W}_n$ or Q does not maximise mutual information for W, leading to a contradiction. See the appendix for a full proof.

5. Bounds on the possible error

To be sure of our results we need to know how close our estimate of capacity is to the real value. We can find such a bound in two ways. First, as we do in this section, we can estimate the error in each of the matrix entries and then calculate the maximum effect that all of these errors might cause on our final result. This method is relatively simple but leads to wide confidence intervals for the final results. A second method is to calculate the distribution that our results come from, in terms of the value we are trying to estimate. This method provides much tighter bounds but, due to the maximising nature of capacity, we must relate our results to a lower bound for capacity: $I(\hat{Q}_m(\hat{W}_n), W)$, rather than the true capacity $I(Q, W)$. While this is a lower bound it is also zero if, and only if, the true capacity is zero:

Lemma 5.1: Let $\hat{W}_n$ be a randomly sampled matrix from n samples and $\hat{Q}_m(\hat{W}_n)$ be the result of m iterations of the Blahut-Arimoto algorithm applied to this matrix, starting from a uniform distribution. Then $I(\hat{Q}_m(\hat{W}_n), W)$ is zero if and only if $C(W)$ is zero.

Each entry in the matrix $\hat{W}_n$ is a single ratio, for each test run with the input i, either the output o is observed, or it is not observed, therefore, by the central limit theorem, $\hat{W}_n(o|i)$ will be normally distributed around the value $W(o|i)$ and we may calculate a confidence interval for this value, for any given certainty. We may extend this to calculate the confidence interval for the biggest error in the whole matrix, the larger the number of entries and the tighter the confidence interval the larger the number of samples needed.

In order to extract the error from the logs in our results we write these errors as ratios, i.e., for a given confidence interval we calculate el_{io} and eg_{io} so that we may be confident that

$$\forall i, o. \hat{W}_n(o|i).el_{io} \leq W(o|i) \leq \hat{W}_n(o|i).eg_{io}$$

and take e_{min} to be the minimum of the el_{io} values and e_{max} to be the maximum of the eg_{io} values. By writing each value of W as a value of W_e and some error we can proved the following lemma:

Lemma 5.2: Given $\hat{W}_n$ estimated from samples of the system W and e_{max} and e_{min}, the largest and smallest error ratios for any entry in the matrix, then for any probability distribution X we have that:

$$e_{min}.(I(X,\hat{W}_n) + log(\tfrac{e_{min}}{e_{max}})) \leq I(X,W) \\ \leq e_{max}.(I(X,\hat{W}_n) + log(\tfrac{e_{max}}{e_{min}}))$$

The proof is given in the appendix. By substituting in $\hat{Q}_m(\hat{W}_n)$ and Q for X we find the following bounds on the true capacity of the system:

Theorem 5.1: Given $\hat{W}_n$, estimated from samples of the system W, and e_{max} and e_{min} and an approximation to the maximising input distribution for $\hat{W}_n$ found using the Blahut-Arimoto algorithm ($\hat{Q}_m(\hat{W}_n)$), such that the maximum error on for the capacity of $\hat{W}_n$ is δ then:

$$e_{min}.(I(\hat{Q}_m(\hat{W}_n),\hat{W}_n) + log(\tfrac{e_{min}}{e_{max}})) \leq C(W) \\ \leq e_{max}.(I(\hat{Q}_m(\hat{W}_n),\hat{W}_n) + \delta + log(\tfrac{e_{max}}{e_{min}})).$$

The proof is given in the appendix. These bounds can be useful when all of the probabilities in the estimated matrix are relatively large and so the error ratios e_{max} and e_{min} are close to 1. However, when any of the matrix values are small, the bounds quickly become too big to be useful and we must turn to the methods described in the following section.

6. The distribution of anonymity

The process of finding our estimation of capacity can be looked on as drawing a value from a distribution. In this section we show that the value comes from a χ^2 distribution if and only if the true capacity is zero and we also find the mean and variance of the distribution if the capacity is non-zero. This allows us to calculate confidence intervals for a bound on the true capacity in terms of our estimated value.

The Blahut-Arimoto algorithm is a numerical process and we do not know the distributions of the results it returns on sampled data. Therefore we relate our estimation of capacity $I(\hat{Q}_m(\hat{W}_n), \hat{W}_n)$ to the mutual information of the true matrix $I(\hat{Q}_m(\hat{W}_n), W)$. We note that $I(\hat{Q}_m(\hat{W}_n), W)$ is zero if and only if $C(W)$ is zero. Therefore, if our estimate is zero then the true capacity is also zero, otherwise we take our estimate to be a close approximation to the true value.

The mean and variance of sampled mutual information has been found in the case that both distributions are unknown ([16], [22]). In our case, we know the input distribution and only sample to find the outputs. Therefore, we first solve the general problem of finding mutual information when the input distribution is known and the matrix is sampled, then we describe how we use this result to test anonymity.

6.1. The distribution of mutual information

Let us denote the input distribution by X and the output distribution by Y. Suppose there are I inputs and J outputs. Let us write $p_i = P(X = i), i = 0, \cdots, I-1$, $p_j = P(Y = j), j = 0, \cdots, J-1$, and $p_{ij} = P(X = i, Y = j)$. In terms of Q and W these are:

- $p_i = Q(i)$=the probability of seeing i,
- $\hat{p}_{j|i} = \hat{W}_n(j|i)$=the estimated transition probability from input i to output j,
- $\hat{p}_{ij} = p_i \times \hat{p}_{j|i}$ =the estimated probability of seeing i and j
- and $\hat{p}_j = \Sigma_i Q(i).W(j|i)$=the estimated probability of seeing j.

The mutual information can then be written:

$$I(X;Y) = \sum_{i=0}^{I-1}\sum_{j=0}^{J-1} p_{ij}\log\left(\frac{p_{ij}}{p_i p_j}\right),$$

and estimated $\hat{I}(X;Y) = \sum_{i=0}^{I-1}\sum_{j=0}^{J-1} \hat{p}_{ij}\log\left(\frac{\hat{p}_{ij}}{\hat{p}_i\hat{p}_j}\right)$, where the $\hat{p}$'s are the relative frequencies of the corresponding states, based on n samples. Also,

$$\hat{p}_i = \sum_{j=0}^{J-1}\hat{p}_{ij} \quad and \quad \hat{p}_j = \sum_{i=0}^{I-1}\hat{p}_{ij}.$$

It may be shown that when the inputs have no relation with the outputs, i.e. $I(X;Y) = 0$, for large n it holds that $2n\hat{I}(X;Y)$ has an approximate χ^2 distribution with $(I-1)(J-1)$ degrees of freedom, see [6]. From that, one may say that $\hat{I}(X;Y)$ has an approximate bias $(I-1)(J-1)/2n$ and approximate variance $(I-1)(J-1)/2n^2$. When $I(X;Y) > 0$, then it may be shown that $\hat{I}(X;Y)$ has mean $I(X;Y) + (I-1)(J-1)/2n + O\left(\frac{1}{n^2}\right)$ and variance

$$\frac{1}{n}\left(\sum_{i,j} p_{ij}\log^2\left(\frac{p_{ij}}{p_i p_j}\right) - \left(\sum_{i,j} p_{ij}\log\left(\frac{p_{ij}}{p_i p_j}\right)\right)^2\right) \\ +O\left(\frac{1}{n^2}\right),$$

see Moddemejer [22]. Brillinger [6] states, that this distribution is normal.

In the case of our anonymity estimate the situation is slightly different in that the input distribution is completely known. Hence, the estimate of $I(X;Y)$ is modified to

$$\hat{I}(X;Y) = \sum_{i=0}^{I-1}\sum_{j=0}^{J-1} \hat{p}_{ij} \log\left(\frac{\hat{p}_{ij}}{p_i \hat{p}_j}\right)$$

There exists no known result that deals with the asymptotic behaviour of mutual information estimates in this situation. In this paper, we develop a distribution of the mutual information estimate for known input distributions when the output is independent of the input, i.e., the mutual information is zero, and then proceed to compute the asymptotic expectation and variance of the mutual information estimate when its actual value is non-zero.

First, for $I(X;Y) = 0$, i.e. X and Y are independent, we have the following;

Theorem 6.1: When X and Y are independent with distribution of X known, for large n, $2n\hat{I}(X;Y)$ has an approximate χ^2 distribution with $(I-1)(J-1)$ degrees of freedom.

The full proof is in the appendix. We note that this theorem implies that if $I(X;Y) = 0$ then $\hat{I}(X;Y)$ is drawn from a distribution with mean $(I-1)(J-1)/2n$ and variance $(I-1)(J-1)/2n^2$.

When $I(X;Y) > 0$, the distribution is no longer χ^2. In this case we have the following result:

Theorem 6.2: When $I(X;Y) > 0$, $\hat{I}(X;Y)$ has mean $I(X;Y) + (I-1)(J-1)/2n + O\left(\frac{1}{n^2}\right)$ and variance

$$\frac{1}{n}\sum_i p_i \left(\sum_j p_{j|i} \log^2\left(\frac{p_{ij}}{p_j}\right) - \left(\sum_j p_{j|i} \log\left(\frac{p_{ij}}{p_j}\right)\right)^2\right) + O\left(\frac{1}{n^2}\right)$$

To prove this we rewrite our estimate as:

$$\hat{I}(X,Y) = H(X) + \hat{H}(Y) - \hat{H}(X,Y)$$

Where $\hat{H}$ is the entropy calculated from the sampled data. As the distribution X is known we know $H(X)$ exactly. We proceed by taking the Taylor expansion of $\hat{H}(Y)$ and $\hat{H}(X,Y)$ to the order of $O(n^{-2})$. This gives us their expected values in terms of the powers of the expected differences between the entries of the probability transition matrix and their true values. As the rows of the matrix are multinomials we know these expectations (see e.g. [22]). Then, from the expected values of $\hat{H}(Y)$ and $\hat{H}(X,Y)$, we find the expected value of $\hat{I}(X,Y)$.

To find the variance we observe that:

$$V(\hat{I}_{XY}) = V(\hat{H}(X,Y)) + V(\hat{H}(Y)) - 2Cov(\hat{H}(X,Y), \hat{H}(Y))$$

As above we find the variance of $\hat{H}_{XY}$ and $\hat{H}_Y$, and their co-variance from the Taylor expansion and the expectations of the rows of the matrix. The full proof is given in the appendix. As suggested by Brillinger [6] we have verified experimental that this distribution is approximately normal (see for example Section 4 of the Appendix).

It may be noted that the expression of the primary ($O(n^{-1})$) part of the variance above reduces to zero when X and Y are independent, which is consistent with variance of the estimate in the case that $I(X;Y) = 0$.

Comparing our result with that of Moddemejer [22], one point of interest is that the distribution of the estimate of the mutual information under independence of the input and the output does not change whether we know the input or not, and the expectation always remains the same, but the variance reduces when there is some information contained about the output in the input, i.e., when the capacity is non-zero.

In both the zero and the non-zero cases, we have a bound on the variance:

Lemma 6.1: The variance of the estimates of mutual information in theorems 6.1 and 6.2 are bound above by $I.J/n$ where I and J are the sizes of the distributions' domains and n is the number of samples used to find the estimate.

This means that taking more samples than the product of the number of inputs and outputs ensures that the variance will be low and the results accurate. The ability to generate this many samples, in a reasonable amount of time, acts as a guide to which systems can and cannot be analysed statistically. We note, however that the variance can actually be much smaller than $I.J/n$ therefore it may also be possible to get a low variance and accurate results with a smaller number of samples.

6.2. Using the distributions for anonymity

Our results on the distribution of mutual information show that the mutual information is zero if, and only if, the distribution of the estimates has mean $(I-1)(J-1)/2n$ and variance $(I-1)(J-1)/2n^2$ (where I is the number of inputs and J the number of outputs). Whereas, the mutual information is non-zero if, and only if, the mean is the true value plus $(I-1)(J-1)/2n$ and the variance is the value given in Theorem 6.2. Therefore our point estimate of information leakage is:

$$max(0, I(\hat{Q}_m(\hat{W}_n), \hat{W}_n) - (I-1)(J-1)/2n).$$

If a single test falls outside the confidence interval for zero mutual information then we may take it as evidence that the capacity is non-zero and calculate the confidence interval accordingly[1]. However, a single test cannot distinguish between zero leakage and a very small amount. If the result is consistent with the χ^2 distribution then we may conclude that the result is between zero and the upper bound of the confidence interval for non-zero mutual information. This leads to the following testing procedure:

A test to estimate information leakage.

1) Fix the secret inputs and observable outputs of the system under test. Ensure that each run of the system is independent.
2) Run n tests of the system with a random input and calculate an estimated matrix $\hat{W}_n$ (to be sure of good results pick $n >> I.J$).
3) Calculate $e = I(\hat{Q}_m(\hat{W}_n), \hat{W}_n)$ and the point estimate for anonymity $pe = max(0, e - (I - 1)(J-1)/2n)$, using enough iterations of the Blahut-Arimoto algorithm to make the error in the capacity of the estimated matrix much smaller than the required accuracy.
4) If $2n$ times e is inside the 95% confidence interval of the $\chi^2((I-1)(J-1))$ distribution then the confidence interval for the capacity is: 0 to $pe+1.65\sqrt{v}$ where v is the variance as given in Theorem 6.2.
5) If $2n$ times the point estimate is outside the 95% confidence interval of the $\chi^2((I-1)(J-1))$ distribution then the confidence interval for the capacity is: $pe - 1.96\sqrt{v}$ to $pe + 1.96\sqrt{v}$ where v is the variance as given in Theorem 6.2.

In many situations a very small leakage would be acceptable, however if we want to be sure of zero leakage then we have to run multiple tests and check the goodness of fit of the variance against the zero and non-zero predictions (tests based on the mean will not be able to distinguish zero and very small mutual information). To check compatibility of the variances we use the test that the observed variance divided by the true variance should be approximately χ^2 with mean one and variance two over the sample size minus one [4]. For very small values of mutual information the variance might be consistent with both predictions, however as the variance of the estimate of values that are truly zero is $O(n^{-2})$ and the variance of the estimate of values that are truly non-zero is $O(n^{-1})$ it will always be possible to distinguish these cases with a large enough n. Therefore, even though for large degrees of freedom a χ^2 distribution will start to resemble a normal distribution, a large enough sample size will always be able to tell the zero and non-zero distributions apart, due to the different orders of magnitude of the variances. This leads to the following test:

A test for zero information leakage.

1) Fix the secret inputs and observable outputs of the system under test. Ensure that each run of the system is independent.
2) Run 100 analyses with sample size n (as described above), to find $\hat{W}_1, \ldots, \hat{W}_{100}$ (theoretically 40 tests should be enough but 100 tests will produce smoother results).
3) Calculate an estimate of the maximising input distribution $Q_e = Q_m(\hat{W}_1)$, then calculate $I(Q_e, \hat{W}_1), \ldots, I(Q_e, \hat{W}_{100})$ and find the variance of these results: v.
4) Calculate the variance predicted by Theorem 6.1 v_{zero} and by Theorem 6.2 $v_{notZero}$.
5) If v/v_{zero} is inside the confidence interval for $\chi^2(2/n)$ and $v/v_{notZero}$ outside the confidence interval then conclude that the information leakage is zero.
6) If v/v_{zero} is outside the confidence interval for $\chi^2(2/n)$ and $v/v_{notZero}$ inside the confidence interval. Then conclude that the information leakage is non-zero.
7) If v is consistent with both predictions then repeat this process with a larger sample size.

We note that, due to the differences in magnitude of the two variance predictions, this test is guaranteed to terminate.

7. Tool support for our analysis method

We have implemented a toolkit to calculate our estimate of anonymity from sampled data, as well as tools to run the tests and collect observations. The aim of this implementation is to allow other researchers to easily calculate anonymity from data they have found themselves. The program and example files are available at http://www.cs.bham.ac.uk/~tpc/AE/.

[1] Here we follow Brillinger and take the non-zero distribution to be normal.

```
payerID = cryptoIDs.randomlyShuffle();    // Shuffle the list of cryptos IDs
// Create all the  Cryptographer object
for (int i = 0; i < noOfCrypt; i++) {
  if (i == payer) {
     cryptographers[i] =
        new Cryptographer(cryptoIDs.get(i), true, coins[i], coins[i+1], ob);
  } else {
     cryptographers[i] =
        new Cryptographer(cryptoIDs.get(i), false, coins[noOfCrypt-1], coins[0], ob);
  }
}
// Start them running
for (int i = 0; i < noOfCrypt; i++) { cryptographers[i].start(); }
```

Figure 3. Part of the code for the master object

Our implementation is roughly 5000 lines of Java code. As part of our implementation, we developed a library that includes methods to efficiently calculate all of the standard information theory values as well as an implementation of the Blahut-Arimoto algorithm. Our tool can also be used to calculate the mutual information for a given input distribution (such as the uniform distribution) and to analyse an exact probability transition matrix, when one can be found, for instance, from a formal model.

The input to the program is a file with a pair of the form $(input, output)$ on each line, indicating a trial of the system under test, started in setup $input$ that resulted in observation $output$. The simplicity of its input is an important advantage of our method. It does not need to know any details about the actual protocol, so it can be applied to any system that can generate a list of input/output pairs. From this list the program estimates each conditional probability $p(o|i)$ as (number of times o seen for i)/(number of times i used as an input). The program then runs the Blahut-Arimoto algorithm until it gets within a specified error, or for a maximum number of iterations. It also automatically calculates the confidence intervals for the results and makes a correction to the estimate, as discussed in Section 6. We extend our software to calculate these distributions and, for multiple samples, to plot the distribution and calculate the goodness of fit (see for example Bickel and Doksum [4]). By default, all results are calculated to a 95% confidence level. The value of capacity can often be more easily understood when it is calculated using log_2 and so becomes the number of bits learnt by the attacker. Therefore, we apply the scaling factor $log_2(e)$ to change the base of the results.

8. Experimental verification of predicated results

In this section, we provide experimental verification of our results by implementing a version of the Dining Cryptographers protocol. This protocol is often proved correct as a benchmark for formal models of anonymity, however, as we show in this section, any real implementation must deal with a number of subtle issues, which may lead to the anonymity being much lower than expected.

Multi-threaded Dining Cryptographers. We have implemented the Dining Cryptographers Protocol in Java, defining classes for:

- a master object that decides who pays and flips the coins using the Java Random library. It sets n concurrent cryptographer objects running. Each cryptographer object is told the coins and whether they pay or not when started.
- n Cryptographer objects that run concurrently using the Java threads library. Each of these objects compares its two coins and then announces “agree” or “disagree”.
- an observer object that listens for, and logs, the output of each Cryptographer as it happens, and if needed starts another run of the system.

One possible way to implement the master class is shown in Figure 3; this randomises the order in which the cryptographers are declared. The result is a package that we can use to run consecutive rounds of the Dining Cryptographers protocol. Each round writes its results to a log with entries which are pairs of the payer ID and ID of the first cryptographer to announce their results and whether they said agree or disagree followed by the results and ID of the second and so on.

no. of tests	5000	10000	50000	100000	200000	400000	True result
Fair coins	0.0032	0.0013	0.0001	0.0000	0.0000	0.0000	0
All bias coins	0.2136	0.1807	0.1759	0.1708	0.1704	0.1705	0.1705

Figure 4. Estimates converging to their true value

```
payerID = cryptoIDs.remove(payer);      // Find and remove the payer's ID
// Create a Cryptographer object for each non-Payer
for (int i = 0; i<(noOfCrypt-1); i++) {
   cryptographers[i] =
      new Cryptographer(cryptoIDs.get(i), false, coins[i], coins[i+1], ob);
}
// Create the Payer
cryptographers[noOfCrypt-1] =
    new Cryptographer(cryptoIDs.get(i), true, coins[noOfCrypt-1], coins[0], ob);
// Start them running
for (int i = 0; i < noOfCrypt; i++) {  cryptographers[i].start(); }
```

Figure 5. Part of the code for the master object II

Results from a single test. We ran tests with our correct implementation of the Dining Cryptographers. First with 3 cryptographers, fair coins and 100,000 samples, which we know should have capacity zero. A typical output of our program was:

```
3 inputs, 4 outputs and 100000 samples
  estimate result = 4.1048E-5
  correction=log2(e)(inputs-1)(outputs-1)
             /2.sampleSize= 4.3280E-5
  This is consistent with the chi^2
  distribution for zero leakage.
Capacity is between 0 and 0.0000
```

Here the sample size is large enough to make the upper bound zero to 4 d.p. although this doesn't rule out a very small non-zero capacity.

Next, we test the same set up, but with biased coins. This has a theoretical capacity of 0.1038, and produces the following results:

```
3 inputs, 4 outputs and 100000 samples
  estimate result = 0.1039
  correction=log2(e)(inputs-1)(outputs-1)
             /2.sampleSize= 4.3280E-5
  This is not consistent with the chi^2
  distribution for zero leakage.
Capacity is between 0.1038 and 0.1039
```

Results from multiple tests. With multiple runs, we can check the goodness of fit of the variance against the zero of non-zero cases. Running 100 tests, we get the following results for the zero case:

```
Results for 100 test of
 3 inputs, 4 outputs and 100000 samples
   observed mean = 4.1051E-5
   observed variance = 5.6186E-10
 correction=log2(e)(inputs-1)(outputs-1)
             /2.sampleSize= 4.3280E-5
 zero variance = 6.2441E-10
 non-zero variance = 1.2657E-6
Capacity is zero
```

The sample size is large enough to clearly distinguish the variances, so we can correctly conclude that the capacity is zero. For the biased coins we get:

```
Results for 100 test of
 3 inputs, 4 outputs and 100000 samples
   observed mean = 0.1038
   observed variance = 3.1746E-6
 correction=log2(e)(inputs-1)(outputs-1)
             /2.sampleSize = 4.3280E-5
 zero variance  = 4.2441E-10
 non-zero variance = 3.071E-6
Capacity is between 0.1038 and 0.1038
```

Here the sample size is large enough to find the correct capacity to 4 d.p. Our tool also produces a visual representation of the distribution of mutual information. Example runs are given in the appendix, and clearly show a normal and χ^2 distribution for the non-zero and zero information leakage, respectively.

Convergence. As the capacity of the Dining Cryptographers for both fair and equally biased coins is known, we can use this implementation to demonstrate our convergence results. Figure 4 shows the results of two sets of tests, the first has all the coins fair and the second has all biased coins that come up heads three times out of four. The true results, found through direct calculation, are shown on the right hand side of the table. We see a clear convergence to the true values, as predicted by Theorem 1.

9. Analysis of a flawed implementation of the Dining Cryptographers protocol

To demonstrate that our method can find faults in a system, we revisit the master object from the Dining Cryptographers example. At first glance the randomisation of the order in which the cryptographers are declared might seem inefficient; we might be tempted to simplify the code by avoiding the ID checks inside the loop and declaring the paying cryptographer last, as shown in Figure 5.

However, analysing 10,000 runs of this implementation of the protocol with 4 cryptographers in our tool tells us that the protocol may be looked on as a channel with capacity of 2 bits. With 4 cryptographers this means that the attacker successfully learns the identity of the payer: this implementation is very broken. Looking at the probability transition matrix constructed by our program from the sampled data starts to tell us why.

Part of this matrix is shown in Figure 6. Here the rows are labelled with the ID of the payer and the columns indicate who said what and when, for instance the column label *0D1A2A3A* corresponds to the cryptographer 0 saying disagree first, followed by 1 then 2 then 3 saying agree. We immediately see that each output implies a particular input. Looking more closely, we see that the payer always announces their results last.

This error is a result of the way in which Java implements concurrent processes on a single CPU machine. The system maintains a queue of processes, executing a single process at a time. The large jobs will be partially executed and then returned to the queue. Jobs are often entered into the queue of processes in the order in which they are declared. Therefore, in our example, the payer will always be entered last and as our jobs are not big enough to make the CPU jump, the payer will always declare last.

This raises the question of what would happen on a multi-core CPU. This will, of course, depend on the exact hardware and operating system used. We repeated our test on a number of different machines and obtained the results given in Figure 7.

The Linux and Apple operating systems seem to preempt their processes more often than the Windows machines and more cores does allow more interleaving. It is clear from our analysis that even on multi-core CPUs, the concurrency of threads does not approximate true concurrency and cannot provide anonymity. The behaviour of this program on these multi-core CPUs is an example of the kind of result that could not be found using a formal model; here it really is necessary to look at trial runs of a system.

Extra work for the payer. Finally, we consider the effect of making the payer perform an actual payment. We alter our code to make the payer open a secure TLS socket connection, as would be required to securely send a credit card number.

The effect this has on the observable outcome of the protocol depends on the speed and type of the hardware. We tested this first on a 900Hz Linux machine with 1 GB of memory, running 100000 tests always returned a capacity of 0.42 bits, to 2 decimal places. Next, we ran the test program on a dual 1.8Gz core Windows machine with 3 GB of memory. In this case, the capacity was significantly higher at 1.42 bits (to 2 d.p.). We speculate that the increased capacity was a result of the faster machine executing the simple non-payers faster, whereas the random number generation and encryption required for the TLS handshake still required significant computation. So, the faster machine will usually finish the simpler tasks first, making the payer stand out.

In each case, the capacity is clearly existent. A leakage of 0.42 bits falls well short of conclusive evidence but would be enough to hint at the identity of the payer. So, while the Dining Cryptographers Protocol is theoretically correct, any attempt by the payer to actually pay the bill may lead to a loss of anonymity.

DC-nets [9] is a proposal for an anonymous communication network built on the Dining Cryptographers protocol. This protocol calls for the sender to hash, and possibly encrypt, the data before sending it. An implementation of DC-nets might leak information in the same way as our example program did, with the hash and encryption causing a noticeable delay to the node sending the message. Unfortunately, there are no implementations of DC-nets that we can test. Such a leak may only cause a small amount of information leakage, and a system designer may well decide that this is acceptable. Our method allows a designer to analyse their system to ensure that the loss of anonymity is within reasonable bounds.

10. Application to the Mixminion remailer

In this section, we show that our method can be used to analyse a node of the Mixminion anonymous

	0D1A2A3A	1D2D3D0A	1A2A3A0D	0D1D3D2A	0A2D3D1D	1A2D3D0D	0A1A3	…
0	0.0	0.1238	0.1260	0.0	0.0	0.1237	0.0	…
1	0.0	0.0	0.0	0.0	0.1227	0.0	0.0	…
2	0.0	0.0	0.0	0.1283	0.0	0.0	0.125	…
3	0.1298	0.0	0.0	0.0	0.0	0.0	0.0	…

Figure 6. Part of the calculated matrix for the Dining Cryptographers

Operating System	1-core CPU	2-core CPU	4-core CPU	8-core CPU
Linux	1.978	n/a	n/a	n/a
Windows	2	1.9963	1.9915	1.9843
Apple	1.9999	1.9731	n/a	n/a

.

Figure 7. Results of running the broken DC implementation on multi-core machines

Message orderings	out A,B,C	out A,C,B	out B,A,C	out B,C,A	out C,A,B	out C,B,A
in 1,2,3	0.0	0.0118	0.0473	0.0118	0.0059	0.9231
in 1,3,2	0.0117	0.0	0.0351	0.0292	0.0	0.924
in 2,1,3	0.005	0.0222	0.0278	0.0444	0.0056	0.8944
in 2,3,1	0.0060	0.012	0.0301	0.0361	0.0060	0.9096
in 3,1,2	0.0067	0.0133	0.04	0.02	0.0067	0.9133
in 3,2,1	0.0061	0.0122	0.0549	0.0244	0.0061	0.8963

Figure 8. Probabilities of the message orderings from Mixminion experiments

remailer. In particular, we show the observable outputs of a single firing of the mix, containing three short messages, leaks no information about the order in which messages entered (this of course doesn't rule out active attacks, or attacks based on correlating traffic across the network).

Mix nodes receive messages and after some interval reorder and forward the messages. Mixminion [13] is an anonymous remailer that uses a network of mix nodes to allow users to send anonymous e-mail. Messages are randomly routed through the network before being sent to their destination. Each node receives messages for a fixed length of time before forwarding all messages together. Some of the messages may be randomly held over until the next firing. The message form also hides information such as the length of the route from the intermediate mixes

Mixes have been the subject of a great deal of research (see [12] for a survey). Much of this work is based on analysing formal or statistical models and uses measures of anonymity that have been specially developed for the system being analysed. What makes our work different is that we are computing a general-purpose definition of anonymity and generating our data from a real, running mix node.

We test whether an observer can link the order of three short messages going into a mix with the messages coming out. These messages were of different lengths and sent to different e-mail addresses. In the different tests, we alternated the order in which the messages entered the mix. So, the secret inputs are the orders in which the three test messages arrive.

To find the observable outputs of the node we ran the WireShark[2] packet sniffer on our test machine. This program recorded all incoming and outgoing packets sent to and from the mix node. To ensure that the observations of the packets leaving our mix were authentic we sent our messages to their destination via real nodes of the Mixminion network[3]. Once all the packets had been collected we recorded the size and number of packets sent to each of the destination mix nodes and the ordering of the packets to each node. These digests of the outgoing streams became the outputs of our channel.

In threshold mode, a mix fires when it has a fixed number of messages and does not hold messages over until the next round. This mix strategy is completely independent between firings, so we used this mode in our tests. While background network traffic and

[2] www.wireshark.org

[3] We only sent messages via nodes where we had received permission from the person running the node, as our test traffic could easily have looked like an attack on the network.

other programs running on the computer may have an effect on the output, we avoid this affecting our results by randomising the order in which the different input message orderings are tested. Therefore, outside conditions will effect all the results equally, and so our experiments fit the requirement of independent and identically-distributed, as described in Section 3.

To gather our test data we ran our own Mixminion node. We set the mix time limit to be 2 minutes, and in each interval sent three known test messages into the node; hence running it as a threshold mix. We found that the mixes would occasionally take longer than the specified interval, so that if we set the interval for less than 2 minutes our test messages would occasionally straddle the boundary between mix firings and so invalidate our results.

We first ran 1000 tests looking only at the ordering of the packets entering and leaving the mix. The results are shown in Figure 8. Message 1 was being send to address A, 2 to B and 3 to C. It was clear that mixminion usually sent the messages out in a fixed order (C then B then A), however occasionally a different order was observed. Was this unusual ordering, or anything else, leaking information on the order of the incoming messages? Or was it unrelated to the Mixminion software and due to the computer's network card, or network conditions? A quick run of our software finds that the capacity of this matrix is 0.0024 which is well within the 95% upper confidence limit for zero leakage (0.0392), therefore there is no evidence of any loss of anonymity.

Next, we ran 10000 tests, in batches of a few hundred, over the course of three weeks and, along with the ordering, also recorded the size and number of packets sent. We disregarded the results when there were large amounts of packet loss due to network disruption; we note that this may be a possible way to attack a mix network. We observed 436 different observable outputs in total. The most common observation by far was 33301 bytes in 32 to 34 packets send to each of the other nodes, with overlapping streams starting in a fixed order. However, occasionally the streams would start in a different order and different numbers of packets, payload size and timings would be observed.

Our software calculated the estimated capacity of the matrix as 0.0249, which is well within the 95% confidence intervals for the χ^2 distribution for the zero case. Leading to a 95% confidence interval for the information leakage as between 0 and 0.0414. Therefore, our result is consistent with a capacity of zero and we may conclude that, in this instance, there is no evidence of any loss of anonymity due to the order in which messages arrive at a Mixminion node. Of course, there are known attacks that target more complicated aspects of the Mixminion system; we plan to investigate whether our method can scale up to detect such attacks in the future.

11. Conclusion

The capacity of a channel with discrete inputs and outputs has been proposed as a metric in a number of areas of computer security. We have shown how these values can be calculated from sampled data and used this theory to calculate the levels of anonymity provided by running programs. Our calculation of the variance of the estimates can also be used to tell when systems are, or are not, too complex to successfully analyse statistically.

As further work, we aim to find the distribution of estimates of conditional mutual information and an upper bound for capacity. For this, we can proceed in the same way as finding the lower bound; for conditional mutual information we can find the Taylor expansions of $H(X|Y)$ and $H(X|Y,Z)$ and for an upper bound on capacity we can find the expansion of $D_x(W \parallel XW)$. This would lead to the mean and variance in terms of the expected differences of the matrix entries, which are known. For conditional mutual information we can use the appropriate adaptation of the Blahut-Arimoto algorithm to find our approximation of the maximising input distributions [15].

The theory in this paper uses a finite number of discrete observations. Using continuous versions of mutual information [24] it may be possible to extend our methods to handle an infinite number of possible observations, as long as they are continuous, e.g., time measurements. This would allow us to analyse systems with a far greater range of observable actions. We may also investigate using stateful channels [11], these are channels with memory, and they would allow us to calculate the capacity of systems where observable outputs may depend on all previous inputs.

References

[1] S. Arimoto. An algorithm for computing the capacity of arbitrary memoryless channels. *IEEE Trans. on Inform. Theory*, IT-18(1):14–20, 1972.

[2] Michael Backes and Boris Köpf. Formally bounding the side-channel leakage in unknown-message attacks. In *ESORICS*, pages 517–532, 2008.

[3] T. Bayes. An essay towards solving a problem in the doctrine of chances. *Philo. Trans. of the Royal Society of London*, 53:370–418, 1774.

[4] Peter J. Bickel and Kjell A. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics*. Prentice Hall, 2006.

[5] R. E. Blahut. Computation of channel capacity and rate distortion functions. *IEEE Trans. on Inform. Theory*, IT-18(4):460–473, 1972.

[6] D. R. Brillinger. Some data analysis using mutual information. *Brazilian Journal of Probability and Statistics*, 18(6):163–183, 2004.

[7] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Prakash Panangaden. Anonymity protocols as noisy channels. *Information and Computation*, 206:378–401, 2008.

[8] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Prakash Panangaden. On the bayes risk in information-hiding protocols. *Journal of Computer Security*, 16(5):531–571, 2008.

[9] David Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1:65–75, 1988.

[10] David Clark, Sebastian Hunt, and Pasquale Malacaria. A static analysis for quantifying information flow in a simple imperative language. *J. Comput. Secur.*, 15(3):321–371, 2007.

[11] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley series in Telecommunications, 1991.

[12] George Danezis and Claudia Diaz. A survey of anonymous communication channels. Technical Report MSR-TR-2008-35, Microsoft Research, January 2008.

[13] George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a type iii anonymous remailer protocol. In *In Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pages 2–15, 2003.

[14] Claudia Díaz, Stefaan Seys, Joris Claessens, and Bart Preneel. Towards measuring anonymity. In Roger Dingledine and Paul F. Syverson, editors, *Proceedings of the workshop on Privacy Enhancing Technologies (PET) 2002*, volume 2482 of *Lecture Notes in Computer Science*, pages 54–68. Springer, 2002.

[15] F. Dupuis, W. Yu, and F. M. J. Willems. Blahut-arimoto algorithms for computing channel capacity and rate-distortion with side information. In *Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on*, pages 179+, 2004.

[16] Marcus Hutter. Distribution of mutual information. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 399–406. MIT Press, 2002.

[17] Pasquale Malacaria and Han Chen. Lagrange multipliers and maximum information leakage in different observational models. In *PLAS '08: Proceedings of the third ACM SIGPLAN workshop on Programming languages and analysis for security*, pages 135–146. ACM, 2008.

[18] Heiko Mantel and Henning Sudbrock. Information-theoretic modeling and analysis of interrupt-related covert channels. In *Pre-Proceedings of the Workshop on Formal Aspects in Security and Trust (FAST)*, 2008.

[19] G. Matz and P. Duhamel. Information geometric formulation and interpretation of accelerated blahut-arimoto-type algorithms. In *Proceedings of the IEEE Information Theory Workshop (ITW)*, pages 66–70, October 2004.

[20] Annabelle McIver and Carroll Morgan. A probabilistic approach to information hiding. pages 441–460, 2003.

[21] Jonathan K. Millen. Covert channel capacity. In *IEEE Symposium on Security and Privacy*, pages 60–66, 1987.

[22] R. Moddemejer. On estimation of entropy and mutual information of continuous distributions. *Signal Processing*, 16:233–248, 1989.

[23] Ira S. Moskowitz, Richard E. Newman, and Paul F. Syverson. Quasi-anonymous channels. In *IASTED CNIS*, pages 126–131, 2003.

[24] Liam Paninski. Estimation of entropy and mutual information. *Neural Comp.*, 15(6):1191–1253, June 2003.

[25] C. R. Rao. *On estimation of entropy and mutual information of continuous distributions*. John Wiles and sons inc., New York, 1965.

[26] Andrei Serjantov and George Danezis. Towards an information theoretic metric for anonymity. In Roger Dingledine and Paul F. Syverson, editors, *Proceedings of the workshop on Privacy Enhancing Technologies (PET) 2002*, volume 2482 of *Lecture Notes in Computer Science*, pages 41–53. Springer, 2002.

[27] P.O. Vontobel, A. Kavcic, D.M. Arnold, and H.-A. Loeliger. A generalization of the blahut-arimoto algorithm to finite-state channels. *IEEE Transactions on Information Theory*, 54:1887 – 1918, 2008.

[28] Ye Zhu and Riccardo Bettati. Anonymity vs. information leakage in anonymity systems. In *Proc. of ICDCS*, pages 514–524. IEEE Computer Society, 2005.

Appendix

1. Proofs

Theorem A.1 (4.1): With enough samples and iterations of the Blahut-Arimoto algorithm our estimate of capacity almost surely tends to the true value:

For any probability p_e and any real number $e > 0$ there exists integers n', m' such that for all $n > n'$ and $m > m'$ and for an estimated probability transition matrix found using n samples $\hat{W}_n$ it holds that

$$p(|I(\hat{Q}_m(\hat{W}_n), \hat{W}_n) - I(Q, W)| > e) < p_e$$

Proof:

We first prove that the capacity of the probability transition matrix, found by sampling, almost surely converges to the capacity of the true probability transition matrix, i.e., $C(W_n) = I(Q(W_n), W_n) \to I(Q, W) = C(W)$ as $n \to \infty$

We prove this by contradiction: assume that there exists some δ such that for all n' there always exists an $n > n'$ such that there is a non-negligible probability that $|C(W) - C(\hat{W}_n)| > \delta$.

Therefore either $C(\hat{W}_n) > \delta + C(W)$ or $C(W) > \delta + C(\hat{W}_n)$.

- In the first case that $C(\hat{W}_n) > \delta + C(W)$

  We know that almost surely $\hat{W}_n \to W$, i.e., for any probability p_w and any real number $e_w > 0$ there exists an n' such that for all $n > n'$ it holds that $p(|W - \hat{W}_n| > e_w) < p_w$ where "$-$" means the maximum difference between any matrix entry.

  We also know that mutual information is continuous in both arguments and finite for a finite number of inputs (see for instance Cover and Thomas [11]) therefore for all $\epsilon > 0$ there exists a m such that for all $n > m$ almost surely:

  $$|I(Q(\hat{W}_n), W_n) - I(Q(\hat{W}_n), W)| \leq \epsilon$$

  Now $I(Q(\hat{W}_n), W_n) = C(\hat{W}_n)$ and $C(\hat{W}_n) > \delta + C(W)$ therefore, because capacity and mutual information are always positive we also have that:

  $$|(\delta + C(W)) - I(Q(\hat{W}_n), W)| \leq \epsilon$$

  We may choose ϵ to be much smaller than δ, so:

  $$I(Q(\hat{W}_n), W) > I(Q, W) = C(W)$$

  but as Q should maximise $I(Q, W)$ this is a contradiction.

- The second case $C(W) > \delta + C(\hat{W}_n)$ is similar:

  The continuity of mutual information tells us that for large enough n and small ϵ almost surely:

  $$|I(Q(W), \hat{W}_n) - I(Q(W), W)| \leq \epsilon$$

  Substituting in for capacity, as above we get:

  $$|I(Q(W), \hat{W}_n) - C(\hat{W}_n) - \delta| \leq \epsilon$$

  and so:

  $$I(Q(W), \hat{W}_n) > C(\hat{W}_n) = I(Q(\hat{W}_n), \hat{W}_n)$$

  Again leading to a contradiction, because $Q(\hat{W}_n)$ should maximise the mutual information for $\hat{W}_n$

Therefore almost surely $C(\hat{W}_n) \to C(W)$ as $n \to \infty$. Further more, we know that the Blahut-Arimoto algorithm ensures that $I(\hat{Q}_m(\hat{W}_n), \hat{W}_n) \to I(Q(\hat{W}_n), \hat{W}_n)$ as $m \to \infty$ (see for instance Vontobel [27]). So we may conclude that $I(\hat{Q}_m(\hat{W}_n), \hat{W}_n) \to C(W)$ as $n, m \to \infty$

□

Lemma A.1 (5.2): Given $\hat{W}_n$ estimated from samples of the system W and e_{max} and e_{min}, the largest and smallest error ratios for any entry in the matrix, then for any probability distribution X we have that:

$$e_{min}.(I(X,\hat{W}_n)+log(\frac{e_{min}}{e_{max}}))\leq I(X,W)\ \leq e_{max}.(I(X,\hat{W}_n)+log(\frac{e_{max}}{e_{min}}))$$

Proof:

We can write the true mutual information in terms of the estimated channel matrix and the (unknown) true error ratios:

$$I(Q,W) \ = \ \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{ao}.log\left(\frac{\hat{W}_n(o|a).e_{ao}}{\Sigma_{a'}Q(a').\hat{W}_n(o|a').e_{a'o}}\right)$$

We expand the log:

$$I(Q,W) \ = \ \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{ao}.(\ log(\hat{W}_n(o|a).e_{ao})-log(\Sigma_{a'}Q(a').\hat{W}_n(o|a').e_{a'o})\ )$$

By definition $\forall a,o.e_{ao}\leq e_{max}$ therefore:

$$I(Q,W) \ \leq \ \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{max}.(\ log(\hat{W}_n(o|a).e_{ao})-log(\Sigma_{a'}Q(a').\hat{W}_n(o|a').e_{a'o})\ )$$

The terms inside the logs are both between 0 and 1, therefore the log values are negative and become more negative as the values inside the log approach 0. So, increasing the term inside the first log and decreasing the second makes the whole term larger.

$$I(Q,W) \ \leq \ \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{max}.(\ log(\hat{W}_n(o|a).e_{max})-log(\Sigma_{a'}Q(a').\hat{W}_n(o|a').e_{min})\ )$$

Now all the e values are the same we can expand the logs again:

$$\begin{aligned}I(Q,W) \ \leq \ & \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{max}.(\ log(\hat{W}_n(o|a))+log(e_{max})\\ & \qquad -log(\Sigma_{a'}Q(a').\hat{W}_n(o|a'))-log(e_{min})\ )\end{aligned}$$

We recombine the logs of the errors and move the e_{max} outside the Σ:

$$I(Q,W) \ \leq \ e_{max}.\Sigma_{a,o}Q(a).\hat{W}_n(o|a).(\ log(\hat{W}_n(o|a))-log(\Sigma_{a'}Q(a').\hat{W}_n(o|a'))+log(e_{max}/e_{min})\ )$$

$$\begin{aligned}I(Q,W) \ \leq \ & e_{max}.(\ \Sigma_{a,o}Q(a).\hat{W}_n(o|a).(\ log(\hat{W}_n(o|a))-log(\Sigma_{a'}Q(a').\hat{W}_n(o|a')))\\ & \qquad +(\Sigma_{a,o}Q(a).\hat{W}_n(o|a)).log(e_{max}/e_{min})\ )\end{aligned}$$

$$I(Q,W) \ \leq \ e_{max}.(I(Q,\hat{W}_n)+(\Sigma_{a,o}Q(a).\hat{W}_n(o|a)).log(e_{max}/e_{min})\ )$$

The term $Q(a).\hat{W}_n(o|a)$ is the probability of getting a and o, so the sum of these, for all possible o and a is 1. Therefore we get:

$$I(Q,W) \ \leq \ e_{max}.(I(Q,\hat{W}_n)+log(e_{max}/e_{min})$$

Working in the other direction:

$$I(Q,W) \ = \ \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{ao}.log\left(\frac{\hat{W}_n(o|a).e_{ao}}{\Sigma_{a'}Q(a').\hat{W}_n(o|a').e_{a'o}}\right)$$

We expand the log:

$$I(Q,W) \ = \ \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{ao}.(\ log(\hat{W}_n(o|a).e_{ao})-log(\Sigma_{a'}Q(a').\hat{W}_n(o|a').e_{a'o})\ )$$

By definition $\forall a,o.e_{min}\leq e_{ao}$ therefore:

$$I(Q,W) \ \geq \ \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{min}.(\ log(\hat{W}_n(o|a).e_{ao})-log(\Sigma_{a'}Q(a').\hat{W}_n(o|a').e_{a'o})\ )$$

The terms inside the logs are both between 0 and 1, therefore the log values are negative and become more negative as the values inside the log approaches 0. So, deceasing terms inside the first log and increasing the second makes the whole term smaller.

$$I(Q,W) \;\geq\; \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{min}.(\; log(\hat{W}_n(o|a).e_{min}) - log(\Sigma_{a'}Q(a').\hat{W}_n(o|a').e_{max})\;)$$

Now all the e values are the same we can expand the logs again:

$$\begin{aligned} I(Q,W) \;\geq\; & \Sigma_{a,o}Q(a).\hat{W}_n(o|a).e_{min}.(\; log(\hat{W}_n(o|a)) + log(e_{min}) \\ & \qquad - log(\Sigma_{a'}Q(a').\hat{W}_n(o|a')) - log(e_{max})\;) \end{aligned}$$

We recombine the logs of the errors and move the e_{min} outside the sigma:

$$I(Q,W) \;\geq\; e_{min}.\Sigma_{a,o}Q(a).\hat{W}_n(o|a).(\; log(\hat{W}_n(o|a)) - log(\Sigma_{a'}Q(a').\hat{W}_n(o|a')) + log(e_{min}/e_{max})\;)$$

$$I(Q,W) \;\geq\; e_{min}.(I(Q,\hat{W}_n) + log(e_{min}/e_{max}).\Sigma_{a,o}Q(a).\hat{W}_n(o|a)\;)$$

The term $Q(a).\hat{W}_n(o|a)$ is the probability of getting getting a and o so the sum of these for all possible o and a is 1.

$$I(Q,W) \;\geq\; e_{min}.(I(Q,\hat{W}_n) + log(e_{min}/e_{max})$$

□

Theorem A.2 (5.1): Given $\hat{W}_n$ estimated from samples of the system W and e_{max} and e_{min} and an approximation to the maximising input distribution for $\hat{W}_n$ found using the Blahut-Arimoto algorithm $\hat{Q}_m(\hat{W}_n)$ such that the maximum error for the capacity of $\hat{W}_n$ is δ then:

$$e_{min}.(I(\hat{Q}_m(\hat{W}_n),\hat{W}_n) + log(\tfrac{e_{min}}{e_{max}})) \leq C(W)\;\; \leq e_{max}.(\hat{Q}_m(\hat{W}_n) + \delta + log(\tfrac{e_{max}}{e_{min}}))$$

Proof: Lemma 5.2 holds when $X = Q$ therefore $C(W) = I(Q,W) \;\; \leq e_{max}.(I(Q,\hat{W}_n) + log(e_{max}/e_{min})$. The capacity of $\hat{W}_n$ is, by definition, equally to or greater than $I(Q,\hat{W}_n)$ therefore $C(W) \;\; \leq e_{max}.(C(\hat{W}_n) + log(e_{max}/e_{min})$ and the limits on the Blahut-Arimoto algorithm insure that $C(\hat{W}_n) \leq I(\hat{Q}_m(\hat{W}_n),\hat{W}_n)+\delta$ therefore $C(W) \leq e_{max}.(\hat{Q}_m(\hat{W}_n) + \delta + log(e_{max}/e_{min})$.

In the other direction, substituting $\hat{Q}_m(\hat{W}_n)$ for X in Lemma 5.2 we get $e_{min}.(I(\hat{Q}_m(\hat{W}_n),\hat{W}_n) + log(e_{min}/e_{max}) \leq I(\hat{Q}_m(\hat{W}_n),W) \leq I(Q,W) = C(W)$. □

2. Proof of Theorem 6.1

Theorem A.3 (6.1): When X and Y are independent with distribution of X known, for large n, $2n\hat{I}(X;Y)$ has an approximate χ^2 distribution with $(I-1)(J-1)$ degrees of freedom.

Proof: The proof follows by observing that the mutual information statistic can be viewed as a loglikelihood ratio statistic for independence of the input and output distribution which have I and J classes respectively, and one category is fixed and controlled. This statistic is in turn approximately equal to

$$\chi^2 = \sum_i \sum_j \frac{(K_{ij} - p_i K_{\cdot j})^2}{p_i K_{\cdot j}}, \tag{2}$$

where $K_{ij} = n\hat{p}_{ij} = np_i\hat{p}_{ji}$ is the number of observations $(X = i, y = j)$ in n samples, and $K_{\cdot j} := \sum_{i=0}^{i-1} K_{ij}$.

Following computations similar to those found in [25], Pg 407, as $(J-1)$ parameters are being estimated, (J of the p_j's are being estimated, but their sum is 1) the quantity χ^2 in (2) has an approximate χ^2 distribution with $(IJ-I)-(J-1) = (I-1)(J-1)$ degrees of freedom. An intuitive explanation of the initial degrees of freedom is that it is the dimension of the subspace for the data, calculated in this case by the number of cells of the matrix of the joint probabilities minus the total number of linear restrictions. There are a total of I linear restrictions because $\sum_{j=0}^{J-1} p_{ij} = p_i$ is known for $i = 0, ...I-1$. Further $J-1$ degrees of freedom are lost by the subsequent estimation of the parameters p_j. □

3. Proof of Theorem 6.2

Theorem A.4 (6.2): When $I(X,Y) > 0$, the estimate $\hat{I}(X,Y)$ has mean $I(X,Y) + (I-1)(J-1)/2n + O\left(\frac{1}{n^2}\right)$ and variance

$$\frac{1}{n}\sum_i p_i \left(\sum_j p_{j|i}\text{log}^2\left(\frac{p_{ij}}{p_j}\right) - \left(\sum_j p_{j|i}\text{log}\left(\frac{p_{ij}}{p_j}\right)\right)^2 \right) + O\left(\frac{1}{n^2}\right),$$

Proof of Theorem 6.2:

Let K_{ij} be the number of observations $(X = i, y = j)$ in n samples. Note that using the fact that p_i is known,

$$K_{ij} = n\hat{p}_{ij} = np_i\hat{p}_{ji}.$$

Then, $\bar{K}_{ij} := E(K_{ij}) = np_{ij} = np_i p_{j|i} = Kp_{j|i}$. We have that $K_{i\cdot} := \sum_{j=0}^{j-1} K_{ij}$ is fixed and is equal to np_i, which is a known number. $K_{\cdot j} := \sum_{i=0}^{i-1} K_{ij}$ has expectation $\bar{K}_j := \sum_{i=o}^{I-1} \bar{K}_{ij}$. Thus, $\bar{K}_j = np_j$. With these definitions, we can write:

$$\hat{I}(X,Y) = H(X) + \hat{H}(Y) - \hat{H}(X,Y), \tag{3}$$

where $H(X)$ is the entropy of X, $H(Y)$ and $H(X,Y)$ similarly defined; $\hat{H}(X,Y) = -\sum_{i,j}\frac{K_{ij}}{n}\log\left(\frac{K_{ij}}{n}\right)$ and $\hat{H}(Y) = -\sum_j \frac{K_{\cdot j}}{n}\log\left(\frac{K_{\cdot j}}{n}\right)$.

Notice that for each i, K_{ij}'s are multinomial with parameters $K_{i\cdot}, p_{0|i}, \cdots, p_{J-1|i}$. Since the $K_{i\cdot}$'s are fixed, for $i \neq i'$, K_{ij} and $K_{i'j'}$ are independent.

Thus, from properties of the multinomial distribution (see [22])

$$V(K_{ij}) = K_{i\cdot}p_{j|i}(1-p_{j|i}),\ Cov(K_{ij}, K_{ij'}) = -K_{i\cdot}p_{j|i}p_{j'|i} \text{ if } j \neq j',\ \text{and } Cov(K_{ij}, K_{i'j'}) = 0 \text{ if } i \neq i',$$

and

$$\begin{aligned} E(K_{ij} - \bar{K}_{ij})^3 &= K_{i\cdot}(2p_{j|i}^3 - 3p_{j|i}^2 + p_{j|i}) = \bar{K}_{ij}(2p_{j|i}^2 - 3p_{j|i} + 1) \\ E(K_{ij} - \bar{K}_{ij})^4 &= K_{i\cdot}^2(p_{j|i}^4 - 2p_{j|i}^3 + p_{j|i}^2) + O(n) = \bar{K}_{ij}^2(p_{j|i}^2 - 2p_{j|i} + 1) + O(n). \end{aligned}$$

Now, using a Taylor expansion of $\hat{H}_{XY} = -\sum_{ij}\frac{K_{ij}}{n}\log\left(\frac{K_{ij}}{n}\right)$ with respect to $\bar{K}_{ij}$, we can write that

$$\hat{H}_{XY} = -\sum_{ij}\frac{\bar{K}_{ij}}{n} - \frac{1}{n}\sum_{ij}\left(1 + \log\left(\frac{\bar{K}_{ij}}{n}\right)\right) - \sum_{ij}\frac{(K_{ij}-\bar{K}_{ij})^2}{n\bar{K}_{ij}} + \sum_{ij}\frac{(K_{ij}-\bar{K}_{ij})^3}{6n\bar{K}_{ij}^2} + O(n^{-2}). \tag{4}$$

Taking expectation of (4), we have

$$\begin{aligned} E(\hat{H}_{XY}) &= -\sum_{ij}\frac{\bar{K}_{ij}}{n}\log\left(\frac{\bar{K}_{ij}}{n}\right) - \sum_{ij}\frac{E(K_{ij}-\bar{K}_{ij})^2}{n\bar{K}_{ij}} + \sum_{ij}\frac{E(K_{ij}-\bar{K}_{ij})^3}{6n\bar{K}_{ij}^2} + O(n^{-2}) \\ &= H_{XY} - \sum_{ij}\frac{(1-p_{j|i})}{2n} + \sum_{ij}\frac{(2p_{j|i}^2 - 3p_{j|i} + 1)}{6n\bar{K}_{ij}} + O(n^{-2}) \\ &= H_{XY} - \frac{I(J-1)}{2n} + O(n^{-2}). \end{aligned} \tag{5}$$

By similar calculations as (5), we get,

$$E(\hat{H}_Y) = H_Y - \frac{J-1}{2n} + O(n^{-2}),$$

which, combined with (5), concludes from from (3) that

$$E(\hat{I}_{XY}) = I_{XY} + \frac{(I-1)(J-1)}{2n} + O(n^{-2}).$$

Now, observe that

$$V(\hat{I}_{XY}) = V(\hat{H}_{XY}) + V(\hat{H}_Y) - 2Cov(\hat{H}_{XY}, \hat{H}_Y). \tag{6}$$

But, from (4)

$$\begin{aligned}
V(\hat{H}_{XY}) &= \frac{1}{n^2} E\left(\sum_{ij}\left(1+\log\left(\frac{\bar{K}_{ij}}{n}\right)\right)(K_{i\cdot j} - \bar{K}_{ij})\right)^2 + O(n^{-2}) \\
&= \frac{1}{n^2}\left[\sum_{ij}\left(1+\log\left(\frac{\bar{K}_{ij}}{n}\right)\right)^2 V(K_{ij}) + \sum_i \sum_{j\neq j'}\left(1+\log\left(\frac{\bar{K}_{ij}}{n}\right)\right)\left(1+\log\left(\frac{\bar{K}_{ij'}}{n}\right)\right) Cov(K_{ij}, K_{ij'})\right] + O(n^{-2}) \\
&= \frac{1}{n^2}\sum_{i,j}\left[\left(1+\log\left(\frac{\bar{K}_{ij}}{n}\right)\right)^2 K_i p_{j|i}(1-p_{j|i}) + \sum_{j'\neq j}\left(1+\log\left(\frac{\bar{K}_{ij}}{n}\right)\right)\left(1+\log\left(\frac{\bar{K}_{ij'}}{n}\right)\right)(-K_i p_{j|i} p_{j'|i})\right] + O(n^{-2}) \\
&= \frac{1}{n^2}\left[\sum_i K_{i\cdot}\left(\sum_j p_{j|i}\left(1+\log\left(\frac{\bar{K}_{ij}}{n}\right)\right)^2 - \sum_j p_{j|i}\left(1+\log\left(\frac{\bar{K}_{ij}}{n}\right)\right)\right)^2\right] + O(n^{-2}) \qquad (7)
\end{aligned}$$

Now, noticing that $K_{i\cdot}/n = p_i$, $\frac{\bar{K}_{ij}}{n} = p_{ij}$,

$$\frac{1}{n^2}(1+\log p_{ij})^2 = \frac{\log^2 p_{ij}}{n^2} + O(n^{-2}) \text{ and} (\sum_j p_{j|i}(1+\log p_{ij}))^2 = (\sum_j p_{j|i}\log p_{ij})^2 + O(n^{-2})$$

we can simplify (7) to

$$V(\hat{H}_{XY}) = \frac{1}{n}\sum_i p_i\left(\sum_j p_{j|i}\log^2 p_{ij} - (\sum_j p_{j|i}\log p_{ij})^2) + O(n^{-2})\right). \tag{8}$$

Using similar calculations as in (8), we get

$$V(\hat{H}_Y) = \frac{1}{n}\sum_i p_i\left(\sum_j p_{j|i}\log^2 p_j - (\sum_j p_{j|i}\log p_j)^2) + O(n^{-2})\right). \tag{9}$$

Finally, we have that

$$\begin{aligned}
Cov(\hat{H}_{XY}, \hat{H}_Y) &= \frac{1}{n^2}\sum_i\sum_{j,j'}\left(1+\log\left(\frac{\bar{K}_{ij}}{n}\right)\right)\left(1+\log\left(\frac{\bar{K}_{\cdot j'}}{n}\right)\right) Cov(K_{ij}, K_{\cdot j'}) + O(n^{-2}) \\
&= \frac{1}{n^2}\sum_i\sum_{j,j'} log(p_{ij}) log(p_j) K_i p_{j|i}(I(j=j') - p_{j'|i}) + O(n^{-2}) \\
&= \frac{1}{n}\sum_i p_i\left(\sum_j p_{j|i}\log p_{ij}\log p_j - (\sum_j p_{j|i}\log p_{ij})(\sum_j p_{j|i}\log p_j)\right) + O(n^{-2}). \qquad (10)
\end{aligned}$$

Combining (8), (9) and (10), we get that

$$
\begin{aligned}
& V(\hat{I}(X,Y)) \\
= \; & \frac{1}{n}\sum_i p_i \sum_j p_{j|i}(\log^2 p_{ij} + \log^2 p_j - 2\log p_{ij}\log p_j) \\
- \; & \frac{1}{n}\sum_i p_i \left((\sum_j p_{j|i}\log p_{ij})^2 + (\sum_j p_{j|i}\log p_j)^2 - 2(\sum_j p_{j|i}\log p_{ij})(\sum_j p_{j|i}\log p_j) \right) + O(n^{-2}) \\
= \; & \frac{1}{n}\sum_i p_i \left(\sum_j p_{j|i}\log^2 \frac{p_{ij}}{p_j} - (\sum_j p_{j|i}\log\frac{p_{ij}}{p_j})^2 \right) + O(n^{-2}).
\end{aligned}
$$

□

4. Program output for the Dining Cryptographers protocol

We have verified our theoretical results by running trials of systems for which we know the true capacity. We give two examples below with plotted distributions, calculated by the tool we developed. The first is of a system that has a capacity of zero and shows a χ^2 distribution. The second is of a system with a non-zero capacity and shows a normal distribution. In both cases, the variance is as predicted.

3 cryptographers, fair coins and 100,000 samples.

```
|     *
|     *
|     *
|     *   *
|     *   *
|     **  *
|   * *** *
|  ** *** *
| ******* * *
| ********* *
| ************
| ************
| ************* *  *
| *******************   *
_______|_______________________________
       M

Results for 100 test of
 3 inputs, 4 outputs and 100000 samples
   observed mean = 4.1051E-5
   observed variance = 5.6186E-10
 correction=log2(e)(inputs-1)(outputs-1)
              /2.sampleSize= 4.3280E-5
 zero variance = 6.2441E-10
 non-zero variance = 1.2657E-6
Capacity is zero
```

3 cryptographers, biased coins and 100,000 samples.

```
                 *
                 * *
                 * *
                 * *   *
                 * *   **
                 * *   **
               * * *  ***
               * * * ****
             * * **********
             ************** **
           * *****************   *
   *     * *******************   * *
_____________________|________________
                     M
```

```
Results for 100 test of
 3 inputs, 4 outputs and 100000 samples
   observed mean = 0.1038
   observed variance = 3.1746E-6
 correction=log2(e)(inputs-1)(outputs-1)
              /2.sampleSize = 4.3280E-5
 zero variance  = 4.2441E-10
 non-zero variance = 3.071E-6
Capacity is between 0.1038 and 0.1038
```