A Formal Characterization of Uniform Peer Sampling based on View Shuffling

Yann Busnel, Roberto Beraldi and Roberto Baldoni

MIDLAB / Università di Roma “La Sapienza” – Italy

MIDLAB Technical Report 04/2009

Abstract

Consider a group of peers, an ideal random peer sampling service should return a peer, which is an unbiased independent random sample of the group. This paper focuses on peer sampling service based on view shuffling, where each peer is equipped with a local view of size $c$. This view should correspond to a uniform random sample of size $c$ of the whole system in order to implement correctly a uniform peer sampling service. To this aim, pairs of peers regularly and continuously swap a part of their local views (shuffling operation). The paper provides a proof that (i) starting from any non-uniform distribution of peers in the peers’ local views, after a sequence of pairwise shuffle operations, each local view eventually represents a uniform sample of size $c$ and (ii) once previous property holds, any successive sequence of shuffle operations does not modify this uniformity property. This paper also presents some numerical results concerning the speed of convergence to uniform samples of the local views.

keywords Peer sampling; Gossip-based protocol; Theoretical analysis; Stochastic process; Numerical evaluation.

1 Introduction

Uniform peer sampling service has been shown recently to be a basic building block for several applications in large-scale distributed systems [8] as information dissemination [1], counting [12], clock synchronization [2], etc. Working on the top of a biased peer sampling can affect either performance, correctness or both of a given application. A sequence of invocations to a peer sampling service returns a sequence of samples of the peers belonging to the system. If samples are unbiased random samples of the system, the peer sampling service is called uniform. There are two main approaches to implement uniform random sampling, random walk and gossip-based protocols.

A random walk on a given graph is a sequential process that consists in visiting the nodes of the graph according to a random order induced by the way the walker is allowed to move. More precisely, the walker moves from one node to one of its neighbors that is selected uniformly at random. The key property of a random walk is that, after a suitable number of steps, called the mixing-time, the visited node is the same as drawn from a uniform distribution [3]. Thus, random walk-based peer sampling mechanisms aim at implementing a biased random walk [12]. Unfortunately, the mixing-time depends on the topological property of the graph, which is generally unknown. Thus, for the reached node to be uniformly sampled, the length of the walk has to be properly tuned. Moreover, this technique may incur in a long delay to return a sample.

This paper focuses on uniform peer sampling based on gossip protocols. We consider a system formed by $n$ peers (i.e., nodes), each provided with a local view of size $c \leq n$. Each node runs a simple shuffling protocol where pair of nodes regularly and continuously swaps part of their local views (shuffle operation).
This protocol is similar to the ones used in [8, 6, 7, 14]. The shuffling protocol aims that local views eventually represent a uniform random sample of the system. The main results presented in this paper show formally that:

1. starting from any non-uniform distribution of nodes in the local views, after a sufficiently long sequence of pairwise shuffle operations executed by the shuffling protocol, each local view represents a uniform random sample of size $c$ among the whole system (Theorem 5.3);

2. once previous property has been established, any sequence of successive shuffle operations does not modify the previous property (Corollary 5.4).

To the best of our knowledge, these results have never been formally proved before, despite the fact that there is empirical evidence shown in many papers [8, 14], that protocols based on view shuffling can provide continuously a uniform sampling.

Let us remark that this result complements the one presented in [4]. Indeed, the authors of [4] propose a protocol based on view shuffling and formally prove that this protocol converges to a uniform peer sampling also in the presence of byzantine peers. Each run of their protocol leads, after a sufficiently long sequence of shuffle operations, to verify the property: “each local view is a uniform random sample of the system”. However, each time a user requires to get a new uniform sample, another instance of this protocol has to be started and it has to converge to a new uniform random sample. Conversely, the shuffling protocol presented in this paper shows that once the local view converges to represent a uniform sample of the system, then successive shuffle operations do not modify the property (Corollary 5.4). Therefore, there is a continuous availability of a uniform random sample without the need to start other instances of the base protocol. The paper finally presents some numerical results related to the shuffling protocol concerning the speed of convergence to uniform samples of the local views.

This paper is organized as follows: Section 2 presents the system model. The shuffling protocol is presented in Section 3 while Section 4 provides an analytical model of the shuffling protocol. Section 5 proves that the local views shaped by the shuffling protocol converge to uniform random samples of the system. Section 6 provide some stochastic evaluations in order to illustrate the best settings according to the system parameters. Finally, related works and conclusion are given respectively in Section 7 and in Section 8.

2 System model

We consider a finite set of $n$ nodes (with $n \geq 2$), which are uniquely identified through a system-wide identifier (ID). Each node $i$ manages a local partial view of the system, denoted $V_i$ of size $c \leq n$ about all the other nodes in the system, including itself.

The view of node $i$ is modelled as a fixed-size set of binary random variables indicating whenever the identifier $k$ appears in $V_i$ or not:

$$X_i = (X_{1i}, X_{2i}, \ldots, X_{ni})$$

where

$$X_{ki} = \begin{cases} 
1 & \text{if } k \in V_i; \\
0 & \text{otherwise}
\end{cases}$$

The vector $X_i$ is referred as the characteristic vector of the view. A vector $X_i$ is associated with a probability vector

$$P_i = (p_{1i}, p_{2i}, \ldots, p_{ni})$$

where $p_{ki}$ is the probability that $k$ belongs to $V_i$, i.e., $p_{ki} = \mathbb{P}[X_{ki} = 1]$.

The whole system is then modelled as the collection

$$S = (X_1, X_2, \ldots, X_n)$$

of the characteristic vectors corresponding to the nodes’ view. The corresponding set of probability vectors is called a configuration of the system,

$$C = (P_1, P_2, \ldots, P_n).$$
Definition 2.1 A view is uniform if at a random instant of time all IDs appear in this view with the same probability.

Definition 2.2 A system is called uniform if all views are uniform.

In this paper we use the notion of potential function to deal with arbitrary configurations.

Definition 2.3 The local potential function of given probability vector $P$ is

$$h(P) = \max_{p_k \in P} \left\{ p_k - \frac{c}{n} \right\}.$$ 

Definition 2.4 The potential function of configuration $C$ is

$$h(C) = \max_{P_i \in C} \{ h(P_i) \} = \max_{P_i \in C} \max_{p_{ki} \in P_i} \left\{ p_{ki} - \frac{c}{n} \right\}.$$ 

The potential function is a sort distance measure between a generic configuration and the uniform configuration, namely the configuration with all probabilities equal to $\frac{c}{n}$. For such configuration, let us introduce the following lemma. First of all, let consider the following property:

 Property 2.5 The expected size of a view $V_i$ is

$$\mathbb{E} \left[ \sum_{k=1}^{n} X_{ki} \right] = \sum_{k=1}^{n} \mathbb{E}[X_{ki}] = c$$

Lemma 2.6 Let $C$ be a distribution of local views. $h(C)$ is zero $\iff$ $C$ is uniform.

Proof. 

($\Leftarrow$) Let consider the distribution $C$ as uniform. As all the views are uniform, all the probability for a node to appear in any view is the same, so called $\bar{p}$. Thus, from Property 2.5, we have:

$$c = \sum_{k=1}^{n} \mathbb{E}[X_k] = n \cdot \bar{p} \implies \bar{p} = \frac{c}{n},$$

and then, for all nodes, the local potential is zero. So, $h(C) = 0$.

($\Rightarrow$) On the other hand, if the potential function is zero ($h(C) = 0$), then, by definition, the maximum for any probability vector is $\frac{c}{n}$. Thus, from Property 2.5, this also implies that all the probabilities are equals to $\bar{p}$, which is the definition of uniformity (cf. Definition 2.2).

3 The shuffling protocol

We now consider a distributed protocol in which nodes manage their views by performing elementary pairwise shuffle or shuffling operation, denoted as $\diamondsuit$. The notation $i \diamondsuit j$ is used to denote that $i$ performs a shuffling operation with $j$. The effect of an operation is to update the nodes’ view, as detailed later in this section. We then show that the protocol makes the system to converge towards a uniform configuration, namely a configuration with zero potential function.

We assume that two shuffles involving a common node may not take place concurrently. Once a node initiates a shuffle, it will be locked until the operation is terminated.

The shuffling operation

The shuffling operation is the core aspect of the whole protocol. The shuffling protocol consists of applying the shuffling operation repeatedly to pairs of nodes $i, j$ taken at random.

This shuffling operation has one parameter, the shuffle length $l$, and involves two views, say $V_i$ and $V_j$. For the sake of simplicity, we will also use the shuffle ratio, $\gamma = \frac{l}{c}$. The operation $\diamondsuit$ acts as follows.
The view \( V_i \) (resp. \( V_j \)) is split into two random parts. The first part, denoted as \( \ell_i \) (resp. \( \ell_j \)), is the sent view, which is a subset of \( V_i \) (resp. \( V_j \)) of size \( l \). The elements in \( \ell_j \) are added to \( V_i - \ell_i \), and inversely. If the size of this new set, \( V'_i = (V_i - \ell_i) \cup \ell_j \) is lower than \( c \) (this could happen if \( \ell_j \) and \( V_i - \ell_i \) have common elements), then \( l' = c - |V'_i| \) elements are taken from \( \ell_i - \ell_j \) at random and added to \( V'_i \). More formally, the shuffling operation consists of the steps presented in Algorithm 1. In the latter, \( \text{UniRand}(h, V) \) returns a subset of \( h \) elements taken uniformly at random from \( V \). The shuffle operation is symmetric in the sense that node \( j \) acts exactly as node \( i \). Moreover, the two nodes make their decisions about which elements to keep from the sent view, if any, independently from each other. Thus, the probability of a node \( k \) to appear in a view is only determined by the elements in the interacting nodes before the shuffle\(^1\).

Consider the following example. Assume that \( c = 7 \) and \( l = 3 \). Consider then a shuffle between the views \( V_i = \{0, 12, 1, 5, 3, 7, 8\} \) and \( V_j = \{3, 11, 4, 5, 8, 2, 1\} \) with sent subset \( \ell_i = \{3, 7, 8\} \), \( \ell_j = \{8, 2, 1\} \). We then have that the first manipulation:

\[
V'_i = (V_i - \ell_i) \cup \ell_j
\]

produces

\[
V'_i = \{0, 12, 1, 5\} \cup \{8, 2, 1\} = \{0, 12, 1, 5, 8, 2\}
\]

As \( |V'_i| = 6 \) while \( c = 7 \), we need to add some random elements of the set

\[
\ell_i - \ell_j = \{3, 7, 8\} - \{8, 2, 1\} = \{3, 7\}
\]

A remark on system partitioning

Unfortunately, the aforementioned operation can lead to a partition of the system. Indeed, consider a system composed by two distinct clusters which are linked by two edges \( i \rightarrow j \) and \( j \rightarrow k \), where \( i, k \) belongs to the first cluster). Assume that \( l = 1 \). If \( i \) makes a shuffle with \( j \), then it may happen that \( i \) sends \( j \) to \( j \) and \( j \) sends \( k \) back to \( i \). If node \( j \) integrates \( j \) in its view and \( i \) replaces \( j \) with \( k \), then the system remain partitioned in two clusters forever, namely none of the node in the first cluster is aware of nodes in the second cluster and vice versa.

This specific case can be avoid using a tricky mechanism, as in [14]. To guarantee that the system could not become partitioned after a shuffling operation, we drive the choice of the shuffling partner to be in \( \ell_i \) and force the initiator node of the shuffle to send its own ID by replacing the one of the partner. This ensures any shuffling operation results into at least to a link reversal (cf. [14] for more details of the proof).

Our biased shuffling operation uses link swap as a technique for preserving system connectivity. The new operation is presented in Algorithm 2. The initiator node chooses \( l \) nodes in its current view to fill \( \ell'_i \). Then, as explained above, it chooses the partner \( j \) at random and replaces it by its own ID to obtain \( \ell_i \), unless \( i \) already appears in \( \ell'_i \) (this is required to keep the view size constant).

### 4 Protocol analysis

In this section, we derive an analytical model of the shuffling protocol, which captures the variation of the system configuration over time. The main symbols used in this paper are reported in Table 1.

---

\(^1\)A correlation would arise if, for example, node \( i \) decides to add the identifier \( k \) received from \( j \) only if \( j \) promises something else back.
Algorithm 2: Biased shuffling operation

```
node i
ℓ_i' ← UniRand(l, V_i)
j ← UniRand(1, ℓ_i')
if i /∈ ℓ_i' then
    ℓ_i ← (ℓ_i' \ {j}) ∪ {i}
else
    ℓ_i ← ℓ_i'

V_i' ← (V_i − ℓ_i) ∪ ℓ_j
V_i ← V_i' ∪ UniRand(c − |V_i'|, ℓ_i − ℓ_j)
```

```
node j
ℓ_j ← UniRand(l, V_j)
V_j' ← (V_j − ℓ_j) ∪ ℓ_i
V_j ← V_j' ∪ UniRand(c − |V_j'|, ℓ_j − ℓ_i)
```

Table 1: List of main symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>Total number of nodes in the system</td>
</tr>
<tr>
<td>c</td>
<td>Size of local view</td>
</tr>
<tr>
<td>l</td>
<td>Size of the sent vector</td>
</tr>
<tr>
<td>γ</td>
<td>Shuffle ratio (γ = \frac{l}{c})</td>
</tr>
<tr>
<td>V_i</td>
<td>Local view of node i</td>
</tr>
<tr>
<td>ℓ_i</td>
<td>Sent view of node i</td>
</tr>
<tr>
<td>X_{ki}</td>
<td>Indication function</td>
</tr>
<tr>
<td>X_i</td>
<td>Characteristic vector of view V_i</td>
</tr>
<tr>
<td>P_i</td>
<td>Probability vector of view V_i</td>
</tr>
<tr>
<td>p</td>
<td>Expected uniform probability (p = \frac{l}{c})</td>
</tr>
<tr>
<td>M_{ij}</td>
<td>Expected number of shared elements between i and j</td>
</tr>
<tr>
<td>i ⊤ j</td>
<td>Suffling operation (i shuffles with j)</td>
</tr>
</tbody>
</table>

4.1 View evolution under the basic shuffling operation

Let consider how the presence of element k in the view of i varies after a shuffling operation among nodes i and j. A shuffling operation between i and j, denoted i ⊤ j, generates two new characteristic vectors, X_i' and X_j', starting from the original vectors X_i and X_j. In other words, after the operation, the view of node i (resp. j) is described by X_i' (resp. X_j').

The evolution of the view over time is then described by a relationship among X and X'. Before describing this relationship, it is important to understand that X_i' is independent from the others random variables X_j'. In fact, the elements that are inserted or removed due to shuffling into the view of node i, are not influenced by elements inserted/removed into the view of node j. In other words, as explained above, i and j do not coordinate somehow about their decisions on the way to change the views. Node i and j act locally and then, independently from each other. Let P_{10} be the probability that, after the shuffle, node k is removed from V_i and P_{01} the probability that k is inserted (for the sake of simplicity indexes are omitted), namely

\[ P_{10} = P[X_{ki} = 0 | X_{ki} = 1] \quad \text{and} \quad P_{01} = P[X_{ki}' = 1 | X_{ki} = 0] \]

The probability that k appears in V_i, given that i ⊤ j, is then

\[ P[X_{ki}' = 1 | i \circ j] = (1 − P[X_{ki} = 1])P_{01} + P[X_{ki} = 1](1 − P_{10}) \]  

This expression has the following meaning. The probability that node k appears in i’s view, after a shuffle between i and j, as given by the probability that k was not in the view and it has been added or the probability that k was already in the view and it has not been deleted. The evolution of a view is best described as a two states Markov chain, see Figure 1, where state 1 (resp. state 0) means that k is (resp. not) in the node i’s view.
Node $i$ receives $l$ elements from $j$. As an element is sent with probability $\gamma$, the expected number of elements that $\ell_j$ and $V_j$ have in common is:

$$\sum_k \mathbb{P}[X_{ki} = 1] \cdot \gamma \cdot \mathbb{P}[X_{kj} = 1] = \gamma \cdot M_{ij}$$

Now, the view size must remain constant. The expected number of elements removed from $i$ must then be equal to the number of new elements added into $V_i$, which is equal to $l - \gamma M_{ij}$. As all elements have to equally likely be removed, the probability to remove the element $k$ is $\frac{l - \gamma M_{ij}}{c}$. From which:

$$P_{10} = \gamma \cdot \left(1 - \frac{\sum_k \mathbb{P}[X_{kj} = 1]}{c} \mathbb{P}[X_{ki} = 1]\right)$$

On the other hand, as element $k$ can be added only if it belongs to $\ell_j$, we have:

$$P_{01} = \gamma \cdot \mathbb{P}[X_{kj} = 1]$$

### 4.2 View evolution under the biased shuffling operation

In this case, we have to distinguish between the initiator and the partner of the shuffle.

#### 4.2.1 Initiator view change

As the partner of an exchange is not forced to put any ID into the sent view, the view evolution of the initiator, which receives the sent view, remains as described above.

#### 4.2.2 Partner point of view

In this case, we have to model the biased part of the shuffle. We have:

$$\mathbb{P}[X'_{ki} = 1|j \circ i] = \begin{cases} 1 & \text{if } k = j \\ (1 - \mathbb{P}[X_{ii} = 1]) \cdot \gamma \cdot \mathbb{P}[X_{jj} = 1] + \mathbb{P}[X_{ii} = 1] \cdot (1 - P_{10}) & \text{if } k = i \\ (1 - \mathbb{P}[X_{ki} = 1])P_{01} + \mathbb{P}[X_{ki} = 1](1 - P_{10}^*) & \text{otherwise.} \end{cases} (2)$$

First, as $j$ sends his own identifier in each exchange that it initiates, we have: $\mathbb{P}[X'_{ji} = 1|j \circ i] = 1$.

If the identifier $j$ is already in $\ell'_j$, then the sent vector contains $i$. Otherwise, $i$ is replaced by $j$ and so $i$ is not sent. Thus, $j$ sends $i$ with probability $\gamma \cdot \mathbb{P}[X_{jj} = 1]$. 
Thus, the probability that after the exchange \(i\) contains its own identifier is the sum of two contributions corresponding to the following events: (i) \(i\) was not present but it was received from \(j\) and (ii) \(i\) was present before the exchange and it was not replaced.

Finally, in all other cases, the value of \(P^*[X'_{ki} = 1 | j \odot i \land k \neq j \land k \neq i]\) is given by the expression given in Equation 1 by adapting \(P_{01}^*\) and \(P_{10}^*\), respectively with \(P^*_0\) and \(P^*_1\). In fact, as one of the slot of \(V_i\) and \(\ell_j\) contains the ID \(j\) for sure, the effective size of \(i\)'s view and the sent vector has to be reduced by one unit.

Thus, we have:

\[
P^*_0 = \frac{l-1}{c-1} \cdot P[X_{kj} = 1]
\]

\[
P^*_{10} = \frac{l-1}{c-1} \cdot \left(1 - \sum_{k \neq j} P[X_{kj} = 1] P[X_{ki} = 1] c^{-1}\right).
\]

### 4.3 Evolution of the system

Let now consider how the system evolves. As explained above, we assume that concurrent operations cannot occur. Thus, we can serialize parallel shuffles in an arbitrary order and assume that only one shuffling operation may take place at a time. Let \(P_{ex}(i, j)\) be the probability that \(i\) and \(j\) make the shuffle, i.e., \(P_{ex}(i, j)\) is the probability that the operation \(i \odot j\) takes place.

As all the nodes ideally initiates an exchange at the same rate, we can consider that the initiator node is selected at random among all the \(n\) nodes. The target node \(j\) is taken at random from \(\ell_i\). Hence

\[
P_{ex}(i, j) = \frac{1}{n} \cdot P[X_{ji} = 1] \cdot \gamma \cdot \frac{1}{l} = \frac{1}{n} \cdot P[X_{ji} = 1] \cdot \frac{1}{c}.
\]

We can describe the global evolution of the system with the following expression:

\[
P[X'_{ki} = 1] = \sum_j P_{ex}(i, j) \cdot P[X_{ki} = 1 | i \odot j] + \sum_j P_{ex}(j, i) \cdot P[X_{ki} = 1 | j \odot i] + \left(1 - \sum_j (P_{ex}(i, j) + P_{ex}(j, i)) \right) \cdot P[X_{ki} = 1]
\]

This last equation means that the probability vector of a node follows the view evolution presented in Equation 1 if it is involved in a view shuffle (Equation 3a and 3b) and remains the same if it is not involved in the last shuffle (Equation 3c).

### 5 Convergence property of the protocol

Let now consider the converge property of the biased shuffling protocol. In particular, we show that if any of the two shuffling protocol is executed by a system with arbitrary view distribution, then eventually the system converges towards a uniform configuration, i.e., a system in which all the local views represent uniform random samples of the system. In order to show this result, we exploit the notion of potential function, introduced in Section 2. We will show that if the potential function of a configuration is greater than zero, then after a shuffling operation the potential function of the configuration is reduced. Roughly speaking, this means that a shuffling operation moves the system towards a “more” uniform system, or makes the system closer to the uniform configuration.

#### 5.1 Basic shuffling operation

Formally, using the first protocol described in Algorithm 1, we have:

**Lemma 5.1 (Operator \(\odot\) reduces the potential)** Let \(C\) and \(C'\) respectively the configuration of the system before and after a basic shuffling operation. Then \(h(C') < h(C)\).
Proof. Let $P$ and $Q$ be two probability vectors of nodes $i$ and $j$ and let $P', Q'$ these vectors after a shuffling operation $i \circ j$. For the sake of simplicity, we denote the maximum probability before the shuffle as $p^* = h(C)$. We prove below that $\forall k \in [1..n]$, (1) $\Delta p_k = p_k' - p^* < 0$, and that (2) $\Delta q_k = q_k' - p^* < 0$ where $p_k, q_k, p_k'$ and $q_k'$ denote respectively $P[k], Q[k], P'[k]$ and $Q'[k]$. This means that the highest probability decreases and no other probabilities can become greater than the previous maximum.

(1) We want to prove that $\Delta p_k < 0$. From equation 1, we have:

$$p_k' = (1 - p_k) \cdot \gamma \cdot q_k + p_k \cdot \left(1 - \gamma + \frac{M_{ij}}{c}\right)$$

$$\implies \Delta p_k = p_k' - p^* = p_k + \gamma \left(q_k - q_k \cdot p_k - p_k + p_k \cdot \frac{M_{ij}}{c}\right) - p^*$$

As $M_{ij} = \sum_k p_k \cdot q_k < \sum_k p^* \cdot q_k = p^* \cdot c$ and $\gamma \leq 1$, we have:

$$\Delta p_k \leq q_k - \frac{q_k \cdot p_k}{} + \frac{M_{ij}}{c} - p^*$$

$$< q_k - q_k \cdot p_k + p_k \cdot p^* - p^*$$

$$= q_k (1 - p_k) - p^* (1 - p_k) = (q_k - p^*) \cdot (1 - p_k)$$

$$\leq 0 \quad \text{as } \forall i, \ p_k \leq p^* \leq 1 \text{ and } q_k \leq p^*.$$  

(2) It remains to prove the same upper bound for $Q'$, i.e. $\forall i, \Delta q_k = q_k' - p^* < 0$. According to the Equation 1, by symmetry, we have:

$$\mathbb{P}[X_{ki}' = 1|j \circ i] = q_k + \gamma \left(p_k - p_k \cdot q_k + q_k \cdot \frac{M_{ij}}{c}\right).$$

Thus, following the same reasoning, we obtain that $\forall i, \Delta q_k < 0$.

Therefore, we can conclude that $\max\{h(P'), h(Q')\} < \max\{h(P), h(Q)\} \leq p^*$. Let us denote $S_i = \sum_j P_{ex}(i, j)$ and $S_i' = \sum_j P_{ex}(j, i)$. Then, according to the Equation 3, we obtain:

$$\forall i, k, \quad \mathbb{P}[X_{ki}' = 1] < S_i \cdot p^* + S_i' \cdot p^* + (1 - S_i - S_i') \cdot p^* = p^*.$$ 

Then, $h(C') = \max_{P_i \in C'} h(P_i) < p^* = h(C)$ as claimed.  

5.2 Biased shuffling operation

We are now proving the same lemma for the second operation described in Algorithm 2:

Lemma 5.2 (Operator $\odot$ reduces the potential) Let $C$ and $C'$ respectively the configuration of the system before and after a biased shuffling operation. Then $h(C') < h(C)$.

Proof. Let $P$ and $Q$ be two probability vectors of nodes $i$ and $j$ and let $P', Q'$ these vectors after a shuffling operation. Moreover, $p_k, q_k, p_k'$ and $q_k'$ denote respectively $P[k], Q[k], P'[k]$ and $Q'[k]$.

For the sake of simplicity, we also denote in this proof the maximum probability before the shuffle as $p^* = h(C)$.

Let split the study according to the three members of Equation 3. We prove below that the highest probability decreases and no other probabilities can become greater than the previous maximum.

(1) Consider the Equation 3a. Section 4.2.1 presents the view evolution from the initiator point of view. As the view evolution in this case is exactly the same as in the basic shuffle, lemma 5.1 gives us, for all $k$:

$$\mathbb{P}[X_{ki}' = 1|j \odot i] < p^*$$

(2) Consider now the Equation 3b. As explicitly denoted in Section 4.2.2, we have to split the analysis in 2 cases: (2a) $k \neq i$ and (2b) $k = i$. 

8
2a First consider the case \( k \neq i \). In this case, we have:

\[
\sum_j P_{ex}(j, i) \cdot (\mathbb{P}[X'_{ki} = 1|j \circ i] - p^*)
\]

\[
= P_{ex}(k, i) \cdot (\mathbb{P}[X'_{ki} = 1|k \circ i] - p^*) + \sum_{j \neq k} P_{ex}(j, i) \cdot (\mathbb{P}[X'_{ki} = 1|j \circ i] - p^*)
\]

\[
= \frac{1}{n} \cdot p_k \cdot \frac{1}{c} \cdot (1 - p^*) + \frac{1}{n} \cdot \frac{1}{c} \cdot \sum_{j \neq k} p_j \cdot ((1 - p_k) \cdot P_{01}^* + p_k \cdot (1 - P_{10}^*) - p^*)
\]

\[
< \frac{1}{n} \cdot p_k \cdot \frac{1}{c} \cdot (1 - p^*)
\]

\[
+ \frac{n - 1}{n} \cdot \frac{1}{c} \cdot p^* \cdot \left( (1 - p_k) \cdot \frac{l - 1}{c - 1} \cdot q_k + p_k \cdot (1 - \frac{l - 1}{c - 1} \cdot \frac{M_{ij}}{c - 1}) - p^* \right)
\]

As in the previous proof, with here \( M_{ij} < p^* \cdot (c - 1) \) and \( \frac{l - 1}{c - 1} \leq 1 \), we have:

\[
< \frac{1}{n} \cdot \frac{1}{c} \cdot p^* \cdot (q_k - p^* \cdot p_k - p^*)
\]

\[
< \frac{1}{n} \cdot (1 - p^*) \cdot \left( q_k - p^* \right) \cdot (1 - p_k)
\]

Thus, the latter expression is lower than 0 if \( \frac{1}{n} + \left( 1 - \frac{1}{n} \right) \cdot (p_k - p^*) \leq 0 \). This inequality is equivalent to \( \frac{1}{n} \cdot (1 + p^* - p_k) \leq p^* - p_k \iff \frac{1}{n} \leq 1 - \frac{1}{1 + p^* - p_k} \leq 1 - \frac{1}{2} = \frac{1}{2} \).

Hence, as \( n \geq 2 \) by definition, we obtain that:

\[
k \neq i \Rightarrow \sum_j P_{ex}(j, i) \cdot (\mathbb{P}[X'_{ki} = 1|j \circ i] - p^*) < 0.
\]

2b Let consider the case \( k = i \). Here, we have:

\[
\mathbb{P}[X'_{ii} = 1|j \circ i] - p^* = (1 - p_i) \cdot \gamma \cdot q_j + p_i \cdot \left( 1 - \gamma \cdot \frac{M_{ij}}{c} \right) - p^*
\]

\[
< q_j + p_i \cdot q_j + p_i - p_i - p_i \cdot p^* - p^* \quad \text{as} \quad \gamma \leq 1
\]

\[
= (q_j - p^*) \cdot (1 - p_i) < 0
\]

Then, we obtain that, for all \( k \):

\[
\sum_j P_{ex}(j, i) \cdot \mathbb{P}[X'_{ki} = 1|j \circ i] < \sum_j P_{ex}(j, i) \cdot p^*
\]

(3) Finally, consider now the Equation 3c. In this part, we simply have to apply the assumption:

\[
\mathbb{P}[X_{ki} = 1] < p^*
\]

Let us denote \( S_i = \sum_j P_{ex}(i, j) \) and \( S'_i = \sum_j P_{ex}(j, i) \). Then, according to the Equation 3, we obtain:

\[
\forall i, k, \quad \mathbb{P}[X'_{ki} = 1] < S_i \cdot p^* + S'_i \cdot p^* + (1 - S_i - S'_i) \cdot p^* = p^*.
\]

Then, \( h(C') = \max_{P_i \in C'} h(P_i) < p^* = h(C) \) as claimed. \( \square \)

5.3 Convergence of both protocols

We are now in the position to state the following theorem, independently of the shuffling operation considered:
Theorem 5.3 (Convergence to uniformity)
Let \( i \) be the number of shuffling operations executed on a system of \( n \) nodes, \( C_0 \) be any initial unpartitioned distribution of local views and \( C_i \) be the configuration of the system after those \( i \) shuffling operations. Local views built by the shuffling protocol presented in Section 3 will converge to uniform random samples of the system, i.e.,

\[
\forall C_0, \lim_{i \to \infty} h(C_i) = 0;
\]

Proof. The claim follows from result comes from Lemma 5.1 and 5.2, as a shuffling operation strictly reduce the global potential, independently of the pair involved in the shuffle. Thus, the distance of the current distribution of sample with the uniformity could only monotonically reduce, due to Equation 3. Then, the distribution of the samples converges to the uniform one due to Lemma 2.6. \( \square \)

Let us now show a corollary stating that once local views represent uniform samples of the system, the shuffling protocol keeps this property true forever.

Corollary 5.4 (Operator \( \diamond \) preserves uniformity) Let \( C \) be a uniform unpartitioned distribution of local views. A shuffling operation executed by the shuffling protocol presented in Section 3 between any pair of two local views \( X_i \) and \( X_j \) belonging to \( C \) produces a distribution \( C' \) that is uniform.

Proof. Lemma 5.1 or 5.2 gives us that the potential of two views involved in a shuffling operation can only decrease. Given the fact that \( C \) corresponds to the uniform distribution, \( X_i \) and \( X_j \) are uniform and \( P_i = P_j \) are vectors with all elements equal to \( \frac{1}{n} \). Thus, due to Lemma 2.6 and Definition 2.4, the potential of \( X_i \) and \( X_j \) are \( h(P_i) = h(P_j) = 0 \). From Lemma 5.1 or 5.2, after the shuffle, \( h(P_i) \) and \( h(P_j) \) cannot increase and thus, remain to 0. Then, \( C' \) is the uniform distribution. \( \square \)

Appendix contains the formal verification of previous corollary.

6 Numerical results

In this section, we apply the analytical model (cf. Equation 3) in order to numerically derive some representative evolutions of a system, in which shuffles are organized into cycles. One cycle corresponds to all nodes initiate exactly one shuffle with a random partner chosen from its own view\(^2\).

Consider a system with view size \( c = 20 \). Initially, the views of nodes are set to \([1..20]\). This corresponds to one of the worst cases of starting state. Indeed, among a population of 100 nodes, the identifiers \([21..100]\) do not appeared in any view at starting point. They will be introduced progressively by the initiator using the biased shuffling operation, as explained in Section 3.

Figure 2 shows the view evolution of one node. The z-axis shows the probability that an ID appears in the view of this node. At the beginning of an execution, the overlay is then fixed: \( c \) nodes have a probability equals to 1 to appear in a view and the other nodes a probability equals to 0. When the protocol runs, all nodes are proceeding to theirs shuffles during each cycle. Figures 2 shows the evolution of each probability \( P[X_{ji} = 1] \) according to the node identifier \( j \) and the iteration of the algorithm, where one iteration corresponds to one gossip cycle. Figures 3 represents the same data but in a planar view (all the probabilities of each time are mapped vertically). Thus, the latter figure shows the evolution of the maximum (and \( a \text{for} \text{ti} \text{ori} \) the minimum) probability value at a given time. It is possible to observe that all the probabilities converge to the average value (\( \frac{1}{n} = 0.2 \)) in less than 40 gossip cycles.

In other hand, we want to focus on another metric which must reach also uniformity. In fact, by simulate the biased algorithm from the same initial system state, we observed the duration of each inter-inview time (IIT); this correspond to the length of the period between two times a given node belongs to the view of a specific node. In a uniformly random behavior, this IIT must be equal to \( \frac{2}{n} \). Figure 4 presents, for a view equals to 10 and different sent view sizes, the evolution of the IIT according to time. One can observe on this figure that the IIT oscillate around very different values just after the starting time (due to the unbalanced representation of each node in all view, in the aforementioned initial state). However, we could observe that the protocol makes these IIT converge to 0.1, which is exactly \( \frac{4}{n} \) in our settings.

\(^2\)This cycle-based behavior is well-known in gossip-based protocols [8, 6, 7, 14].
Figure 2: Evolution of $\mathbb{P}[X_{ji} = 1]$ for $i$ fixed according to the gossip cycle iteration.  
*Settings: $n = 100$, $c = 20$ and $l = 4$ for 50 iterations.*

Figure 3: Evolution of $\mathbb{P}[X_{ji} = 1]$ for $i$ fixed according to the gossip cycle iteration (Planar view).  
*Settings: $n = 100$, $c = 20$ and $l = 4$ for 50 iterations.*
Figure 4: Duration of presence in a view $V$ of a node, initially located or not in all views, for different $\ell$ size.

Figure 5: Duration of presence in a view $V$ of a node, initially located or not in all views, for different $\ell$ size.
For the comparison purpose, we model the evolution of the system according to Equation 3, for both basic and biased shuffling operation. As in Figure 3, but according to time and not to cycle iteration, we present the probability vector evolution of one view, in Figure 5. The latter speaks of the equivalence of both approaches in term of convergence speed. Moreover, we can observe that the biased operation could lead to decrease a bit faster the very high probability. This is due to the fact that each shuffle initiator node input its own ID in the sent view, and then, refresh the knowledge of its ID faster than the basic approach.

Last but not least, in order to evaluate the impact of the system parameters, Figure 6 presents the average convergence time required to reach the uniform sampling, according to the ratio $\gamma = \frac{l}{c}$, for different settings of the system (from 100 to 1,000 nodes with a view size varying from 10 to 20), starting from the same aforementioned worst case. This figure speaks about how to obtain the best convergence time according to $\gamma$. Independently of the size of the network, the size of the view $c$ and the initial state, the fastest convergence is obtained with a ratio $\gamma = 0.5$ (represented by a vertical line on Figure 6). Thus, in the design of a gossip-based protocol, $l$ has to be set to the half of $c$ in order to obtain the highest efficiency in term of convergence speed.

This conjecture can be intuitively shown as sketched below. A shuffle operation with a sent vector $\ell$ between two nodes is equivalent to a shuffle with the complementary of $\ell$ (i.e. $V - \ell$), followed by swapping the ID of these two nodes (cf. Figure 7). Indeed, in this figure, the content of $V_i$ after the shuffle on the left side is equivalent to the content of $V_i$ on the right side after (1) a shuffle with the sent vector $\ell'_i = V_i - \ell_i$ and (2) swapping the node’s ID ($i$ becomes $k$ and vice versa).

Now, consider $l \leq \frac{c}{2}$. It is obvious that the higher the size of the sent vector, the greater the effectiveness\(^3\) of a shuffle. Moreover, according to the above equivalence, a shuffle with $l$ is equivalent to a shuffle with $c - l$. Thus, for $l \geq \frac{c}{2}$, the lesser the size of the sent vector, the greater the effectiveness of a shuffle. So, the greatest effectiveness is reached for $l = \lfloor \frac{c}{2} \rfloor$, as confirmed numerically in Figure 6.

Let us proved it formally, in Theorem 6.2 below. First, we have to express in a measure of the effectiveness:

**Definition 6.1 (Shuffling Effectiveness)** The effectiveness of a shuffle operation correspond to the magnitude of difference between an update view and both of the views before the shuffle, i.e.:

$$E(P') = \min\{P \cdot P', Q \cdot P'\}$$

\(^3\)Roughly speaking, effectiveness represents how different the shuffled views are from the ones before the shuffle. The higher the difference, the greater the effectiveness.
Figure 7: Intuitive equivalence between a small $\ell$ value and the opposite $c - \ell$ ones.

where $P \cdot P'$ represent the scalar product between the vectors $P$ and $P'$.

Indeed, according to the reasoning above, the core idea of a shuffling operation is to mix at most as possible both views involved in the shuffle. Thus, more different $P'$ is from its initial state $P$ is a good measure, but it has also to be balance with its similarity to the partner ones $Q$.

Starting from this definition, we should maximize the effectiveness of each operation. We prove below that this maximization is achieved for $\gamma = 0.5$.

**Theorem 6.2 (Greatest Effectiveness of a Shuffle)** Given two probability vector $P$ and $Q$. The maximum value of the expected effectiveness is reach for $\gamma = \frac{1}{2}$.

**Proof.** Consider that $P$ (resp. $Q$) is the probability vector of node $i$ (resp. $j$). Let us remark that $P \cdot Q = \sum_k p_k \cdot q_k = M_{ij}$. Then, $P \cdot Q$ represents the expected number of elements in common in $V_i$ and $V_j$.

Moreover, the number of new element added in $V_i$ after a shuffle corresponds to the size of $\ell_j$ the partner’s sent view, minus the numbers of element in this sent view which was yet in $V_i$. As we show in Section 4 that $|V_i \cap \ell_j| = \gamma \cdot M_{ij}$, this number of new elements inserted is given by:

$$l - \frac{l}{c} P \cdot Q$$

Thus, we can infer an expression of $P \cdot P'$ and $Q \cdot P'$. In first hand, $P \cdot P'$ corresponds to the number of element in common between $P$ and $P'$, which is the size of the view minus the number of new element inserted. In other hand, $Q \cdot P'$ corresponds to the number of element in common between $Q$ and $P'$, which is the size of the sent view (inserted for sure in $V_i$ after the shuffle) plus the unsent elements that was in common with $P$ and $Q$. Then, we obtain:

$$\begin{align*}
P \cdot P' &= c - l + \frac{l}{c} \times P \cdot Q = l \left(\frac{P \cdot Q}{c} - 1\right) + c \\
Q \cdot P' &= l + \left(1 - \frac{l}{c}\right) P \cdot Q = l \left(1 - \frac{P \cdot Q}{c}\right) + P \cdot Q
\end{align*}$$

In the proof of Lemma 5.1, we prove that $M_{ij} \leq c$. So, we have $\frac{P \cdot Q}{c} \leq 1$. Given a fixed view size $c$, $P \cdot P'$ (resp. $Q \cdot P'$) is then a linear function of $l$ with a negative (resp. positive) slope, and a range equals from $0$ to $P \cdot Q$ (resp. from $P \cdot Q$ to $0$) cf. Figure 8. Then, the effectiveness is equal to $P \cdot P'$ for small
value of \( l \) and to \( Q \cdot P' \) for large value of \( l \). Thus, greatest value of \( \mathcal{E} \) is obtained for:

\[
P \cdot P' = Q \cdot P' \iff l \left( \frac{P \cdot Q}{c} - 1 \right) + c = l \left( 1 - \frac{P \cdot Q}{c} \right) + P \cdot Q
\]

\[
\iff l \left( 2 \cdot \frac{P \cdot Q}{c} - 2 \right) = P \cdot Q - c
\]

\[
\iff 2 \cdot l \cdot \frac{P \cdot Q - c}{c} = P \cdot Q - c
\]

\[
\iff l = \frac{c}{2}.
\]

This claim that the greatest effectiveness is reach for \( \gamma = \frac{1}{2} \).

\[\square\]

7 Related works

Apart of the paper presented in [4] that we discussed in Section 1 and that remains the one that presents the result closest to ours, several contributions have been proposed in the context of gossip-based peer sampling service [8, 14]. In these works, authors propose and study the same framework than the ones we modelled in this paper (as known as gossip-based protocol). Although, the evaluation of their samples’ distribution is conducted only using empirical experimentations. To the best of our knowledge, no fully theoretical analysis of the shuffling protocol with respect to sampling uniformity has been proposed so far.

Several contributions provided some fully theoretical analysis of gossip-based protocols as [8, 9, 10, 5]. However, those analysis aims to provide some theoretical outcomes on a specific characteristic of these protocols as convergence speed of dissemination protocols, by defining precise lower and upper bound of the mixing time, degree balancing, etc. Nevertheless, in these works, authors do not consider the local view as the information to analyse. In their works, the network is modelled as a probabilistic matrix, which represents the meeting probability of any pair of peers, and this matrix is used as a building block of their analyses. Our study can then be used to provide this specific matrix and/or to confirm that the matrix used in these related works are consistent with the real behavior of gossip-based protocols.

As remarked in Section 1, random walks have been also used to provide uniform peer sampling [12, 15]. These contributions proposed how to bias the simple random walks model in the way to extract uniform sampling. Both of them provide a theoretical analysis of their protocols. Finally, a solution of the peer sampling service, based on a structured P2P system, has been proposed in [11]. Authors propose an algorithm based on Chord [13] and proved that it provides nodes with uniform random samples of the system.
8 Concluding Remarks

The paper has provided a theoretical ground to the fact that a shuffling protocol provides eventually nodes with uniform random samples of a system. Before this was only an empirical evidence. Differently from [4], our analysis shows that the same instance of the shuffling protocol can provide permanently a node with uniform sample of the system. Corollary 1 formally grasps this difference. The paper also presented a numerical evaluation of the shuffling algorithm on its convergence speed of the local views to uniform random samples and also what is the best fraction of the local views to swap in a shuffling operation to get best convergence speed.

Acknowledgment We would like to warmly thank Leonardo Querzoni for his help with the simulations.

References


Appendix

The result expressed in Corollary 5.4 can also be verified formally.

**Corollary 8.1 (Operator \(\circ\) preserves uniformity)** Let \(C\) be a uniform unpartitioned distribution of local views. A shuffling operation executed by the shuffling protocol presented in Section 3 between any pair of two local views \(X_i\) and \(X_j\) belonging to \(C\) produces a distribution \(C'\) that is uniform.

**Proof.** From the hypothesis on \(C\), we have \(\forall k, P[X_{ki} = 1] = \frac{c}{n}\). Hence:

\[
P_{01} = \frac{l}{c} \times \frac{c}{n} = \frac{l}{n} \quad \text{and} \quad P_{10} = \frac{l}{c} \cdot \left( 1 - \frac{\sum j \frac{c}{n}}{c} \right) = \frac{l}{c} \cdot \left( 1 - \frac{c}{n} \right)
\]

With these values, from Equation 1, we have:

\[
P[X_{ki}' = 1|i \circ j] = \left( 1 - \frac{c}{n} \right) \frac{l}{n} + \frac{c}{n} \left( 1 - \frac{l}{c} \cdot \left( 1 - \frac{c}{n} \right) \right) = \frac{c}{n}.
\]

\(\square\)