Service d'échantillonage uniforme résiliant aux comportments malveillants

Emmanuelle Anceaume, Yann Busnel, Bruno Sericola

Service d'échantillonage uniforme résiliant aux comportments malveillants

Emmanuelle Anceaume[1], Yann Busnel[2], Bruno Sericola[3]

[1]*IRISA & CNRS, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France*
[2]*LINA & Université de Nantes, 2 rue de la Houssinière, BP 92208, 44322 Nantes Cedex 03, France*
[3]*INRIA Rennes – Bretagne Atlantique, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France*

Nous proposons une solution au problème d'échantillonage uniforme dans les systèmes à grande échelle en présence de comportements byzantins. Notre premier algorithme permet d'uniformiser à la volée un flux de données (items) de taille non bornée, sous l'hypothèse que les probabilités exactes d'occurrence des items sont connues. Nous modélisons le comportement de notre algorithme par une chaîne de Markov dont nous étudions le régime stationnaire et le transitoire. Notre second algorithme relache l'hypothèse de connaissance de la probabilité d'occurrence des items dans le flux initial. Ces probabilités sont estimées à la volée en utilisant un espace mémoire logarithmique en la taille du flux. Nous évaluons la résilience de cet algorithme face à des attaques ciblées et par innondation. Nous quantifions l'effort que doit fournir l'adversaire (*i.e.*, nombre d'items à injecter dans le flux initial) pour violer la propriété d'uniformité.

Keywords: Echantillonnage uniforme, flux de données, adversaire byzantin, algorithme d'approximation probabiliste.

1 Introduction

The uniform node sampling service offers a single simple primitive to applications using it, which returns the identifier of a random node that belongs to the system. Providing at any time randomly chosen nodes in the system has deserved a lot of attention to construct large scale distributed applications. Node sampling is a cooperative service in the sense that all the nodes of the system contribute to this service by continuously sending and forwarding information about their presence. Unfortunately, the unavoidable presence of malicious nodes in large scale and open systems seriously impedes the construction of uniform node sampling. The objective of malicious nodes mainly consists in continuously and largely biasing the input data stream out of which samples are obtained, to prevent (correct) nodes from being selected as samples. Consequences of these collective attacks (also called Sybil attacks) are, among others, the overwhelming load of some specific nodes when it is used to provide random locations for data caching or storage, or the eventual partitioning of the system when the node sampling service is used to build nodes local views in epidemic-based protocols. Solutions that basically consist in storing the identifier of all the nodes of the system so that each of these node identifiers can be randomly selected when needed are impracticable and even infeasible due to the size and the dynamicity of such networks. Rather providing a solution that requires as little space as possible (*e.g.*, sublinear in the population size of the system) is definitely desirable. Bortnikov *et al.* [BGK+09] have recently proposed a uniform node sampling algorithm that tolerates malicious nodes by exploiting the properties offered by min-wise permutations. The sampling component outputs the node identifier whose image value under the randomly chosen permutation is the smallest value ever encountered. Thus eventually, by the property of min-wise permutation, the sampler converges towards a random sample. However by the very same properties of min-wise permutation functions, once the convergence has been reached, it is stuck to this convergence value independently from any subsequent input values. Thus the sample does not evolve according to the current composition of the system, which makes it static.

In this paper, we address this problem by first proposing an omniscient algorithm capable of tolerating any bias introduced by the adversary in the input stream. By omniscient we mean that the algorithm knows

the number of occurrences of each received element in the full input stream. We analyze the stationary and transient behaviour of this algorithm through a Markov chain analysis. We then propose a randomized approximation algorithm capable of outputting an unbiased and non static sample of the population whatever the strategy of the adversary is. This sample may deviate from an exact uniform sample, however the deviation is bounded with any tunable probability. This algorithm is a one-pass algorithm and only compact synopses or sketches that contain the most important information about data items are locally stored. This algorithm does not require any a priori knowledge neither on the size of the input stream, nor on the number of distinct elements that compose it, nor on the frequency distribution of these elements. We then evaluate the minimum effort that needs to be exerted by a strong adversary to bias the output stream when two representative attacks are launched, *i.e.*, the *targeted attacks* in which the adversary focuses on biasing the frequency of a single node identifier, and the *flooding attack* which aims at biasing all the node identifiers frequencies. One of the main results of this analysis is the fact that the effort that needs to be exerted by the adversary to subvert the sampling service can be made arbitrarily large by any correct node by just increasing the memory space of the sampler. Finally, extensive simulations (both on real data and synthetic traces) confirm the robustness of our sampler service. To the best of our knowledge, no previous work has proposed such an analysis.

2 System model

We consider a large scale and dynamic open system $\mathcal{N}$ in which each node $i \in \mathcal{N}$ receives a very large stream σ_i made of node identifiers (also denoted ids). We denote $n = |\mathcal{N}|$. Node identifiers arrive quickly and sequentially. Each node identifier j of σ_i is drawn from a set $\Omega = \{1, \ldots, 2^r\}$, where r is chosen to be large enough to make the probability of identifier collision negligible. The number of times a node identifier j recurs in the stream is called the *frequency* of j. For memory constraints, nodes can locally store only a small amount of information with respect to the number of ids in the system. Thus the stream needs to be processed in an online manner, *i.e.*, any item of the stream that has not been locally stored for any further processing cannot be read any more. In addition the amount of computation per data element of the stream must be low to keep pace with the stream.

We assume the presence of malicious nodes that collectively try to subvert the system. We model these adversarial behaviors through an adversary that fully controls and manipulates these malicious nodes. We suppose that the adversary is strong in the sense that it may actively tamper with the data stream of any node i by observing, and inserting any number of malicious nodes identifiers. Indeed, the goal of the adversary is to judiciously increase the frequency of f chosen node identifiers to bias the sample built by non malicious nodes. The number f is chosen by the adversary and depends on the sampling protocol parameters. Note that each malicious node identifier does not need to correspond to a single real node. Indeed, the adversary will augment its power by generating numerous node identifiers, such that only a limited number of real malicious nodes are linked to these identifiers. However, affecting multiple identifiers to a single node is costly as one needs to interact with a central authority to receive a certificate assessing the validity and integrity of the identifier. A node present in the system that is not malicious is said to be *correct*. Note that correct nodes cannot *a priori* distinguish correct node identifiers from malicious ones. Classically, we assume that the adversary can neither drop a message exchanged between two correct nodes nor tamper with its content without being detected. This is achieved by assuming the existence of a signature scheme (and the corresponding public-key infrastructure) ensuring the authenticity and integrity of messages. This refers to the authenticated Byzantine failure model. We finally suppose that any algorithm run by any correct node to build a uniform node sampling service is public knowledge to avoid some kind of security by obscurity. However the adversary has not access to the local random coins used in the algorithms.

3 Node sampling service tolerant to malicious nodes

3.1 The addressed problem

A node sampling service tolerant to malicious nodes is a functionality local to each correct node i of the system. Although malicious nodes have also access to a sampling service, we cannot impose any as-

sumptions on how they use it as their behavior can be totally arbitrary. This service continuously reads the input stream σ_i received by node i. Data streams are made of the node identifiers exchanged within the system. Note that the analysis presented in this paper is independent from the way data streams are built. That is, they may result from the continuous propagation of node ids through gossip-based algorithms, or from the node ids received during random walks initiated at each node of the system. In addition, the input stream of any correct node can be arbitrarily biased by an adversary, which is achieved by infinitely often augmenting it with the f ids it manipulates. The objective of the sampling service strategy is to process on the fly the input stream and to output a stream guaranteeing both Uniformity and Freshness. Specifically, if $S_i(t)$ denotes the output of the sampling service at any correct node i at any discrete time t, then a sampling service tolerant to malicious behaviors should meet the following two properties.

Property 3.1 (Uniformity) *For any $t \geq 0$, for any node id $j \in \mathcal{N}$, $\mathbb{P}\{S_i(t) = j\} = \frac{1}{n}$.*

Property 3.2 (Freshness) *For any $t \geq 0$, for any node id $j \in \mathcal{N}$, $\{t' > t \mid S_i(t') = j\} \neq \emptyset$ with probability 1.*

Uniformity states that any node in the system should have the same probability to appear in the sample of correct nodes in the overlay, while Freshness says that any node that recurs infinitely often in the stream, should have a non-null probability to appear infinitely often in the sample of any correct nodes in the system.

3.2 An omniscient and a knowledge-free one-pass algorithms

This section first presents an omniscient one-pass algorithm that guarantees both the Uniformity and Freshness properties. By omniscient, we mean that the algorithm knows exactly the occurrence probability p_j of j in the full stream σ_i (Hypothesis $H1$). Note however that the algorithm does not know ahead of time the identifiers that will appear in σ_i. This knowledge is built on the fly when reading σ_i. The omniscient strategy has uniquely access to a data structure Γ_i, referred to as the *sampling memory*, whose cardinality of Γ_i is constant and is denoted by c with $c \ll n$. The sampling memory contains the node ids that are selected by the strategy when reading σ_i. Specifically, the omniscient algorithm reads on the fly and sequentially the input stream and, for each read element j, decides whether j is a good candidate for being stored into the constant size memory Γ_i or not. If p_j is very small, then j must definitively be stored into Γ_i so that j might have a chance to be part of the output stream. On the other hand, with larger p_j, there will be other opportunities for the sampler to receive j in the future. Inserting j into Γ_i with a well chosen probability a_j is a necessary condition to prevent very frequent ids from continuously eclipsing the ids already stored in Γ_i. Although, this is not sufficient to guarantee that a rare id k already stored in Γ_i will not be evicted each time a new id j is stored (assuming that Γ_i is full upon receipt of j). Recall that the goal of the adversary is to prevent identifiers of correct nodes to uniformly appear in the output stream. A sufficient condition is achieved by removing k from Γ_i with probability $r_k / \sum_{\ell \in \Gamma_i} r_\ell$, where $r_1, \ldots, r_n$ are positive real numbers. Finally, a random node id k' is chosen from Γ_i and written in the output stream (note that k' is not removed from Γ_i). We show in the companion paper [ABS13] that setting $a_j = q/p_j$ with $q = \min_{\ell \in \mathcal{N}} p_\ell$ and $r_j = \frac{1}{c}$ guarantees that the algorithm converges to a stationary regime where both Uniformity and Freshness properties hold, and that the time to converge decreases w.r.t. the size c of the sampling memory.

We now show how to extend this algorithm to get rid of hypothesis $H1$. Clearly such an assumption is unrealistic since the adversary may modify on the fly the occurrence probability of any node identifier in the stream by increasing the occurrence frequency of the f node identifiers it manipulates. This extension, called the *knowledge-free algorithm* makes no assumption with respect to the input stream σ_i. For each received j from σ_i, it selects the id that will be part of the output stream by solely relying on an estimation of p_j. Both estimations are computed on the fly by using very few space and a small number of operations. Specifically, the knowledge-free strategy uses one additional data structure with respect to the omniscient one. This data structure is the *Count-Min (CM) Sketch* [CM05]. The CM sketch is built on the fly and provides at any time, and for each j read from σ_i, an approximation of the number of times j has appeared in σ_i from the inception of the stream. The error of the estimator in answering a query for the frequency of j is within a factor of ε with probability δ. Sketch uses a two-dimensional array $\hat{F}$ of $k \times s$ counters with $k = \lceil e/\varepsilon \rceil$ and $s = \lceil \log_2(1/\delta) \rceil$, and a collection of 2-universal hash functions $\{h_1, \ldots, h_s\}$. Each time an item j is read from the input stream, this causes one counter per line to be incremented, *i.e.*, $\hat{F}[v][h_v(j)]$ is

incremented for all $v \in \{1,\ldots,s\}$. When a query is issued to get an estimate of the frequency of j (*i.e.*, the number of occurrences of j read so far from the stream), the returned value corresponds to the minimum among the s values of $\hat{F}[v][h_v(j)]$ ($v \in \{1,\ldots,s\}$). The space required by the CM sketch is proportional to $\frac{1}{\varepsilon}\log_2\frac{1}{\delta}$, and the update time per element is significantly sublinear in the size of the sketch [CM05].

3.3 Performance evaluation

The omniscient algorithm cannot be tampered with any adversary [ABS13]. We have then evaluated the minimum effort that needs to be exerted by a strong adversary to bias the frequency estimator [CM05], when two representative attacks are launched, *i.e.*, the *targeted attacks* in which the adversary focuses on biasing the frequency of a single node identifier, and the *flooding attack* which aims at biasing all the node identifiers frequencies. Both evaluations are conducted by modeling them as an urn problem. One of the main results of this analysis is the fact that the effort that needs to be exerted by the adversary to subvert the sampling service can be made arbitrarily large by any correct node by just increasing the memory space of the sampler. We have implemented both the omniscient and knowledge-free algorithms and have conducted a series of experiments on different types of streams and for different parameters settings [ABS13]. We have fed our algorithms with both real-world data sets and synthetic traces that are representative of over-represented (malicious) node identifiers. Due to space constraints, we present some results that summarize the quality of our algorithms. Figure 1 illustrates the behaviour of both algorithms. It presents a kind of isopleth in which the horizontal axis shows time, the vertical axis represents the node identifiers, and the body of the graph depicts the frequency of each node identifier. A lighter color is representative of a very frequent node identifier. The figure at the top of Figure 1 represents the frequency of each node identifier in the input stream of the node sampler. This figure shows that at the inception of the stream, a few number of node identifiers have been received in the input stream which explains the dark color on the left. As time elapses, the number of received identifiers increases (up to $40,000$), and progressively the bias of the input stream appears : a small number of identifiers recur with a high frequency equal to 400, while the frequency of the other node identifiers is significantly lower. Now the two other figures represent the output of the node sampler run with respectively the knowledge-free strategy and with the omniscient one. Clearly the omniscient strategy succeeds in outputting a uniform stream, illustrated by a color that progressively and uniformly becomes lighter as the number of received identifiers augments. The knowledge-free strategy is not as performant as the omniscient one, nevertheless it succeeds in significantly decreasing the peak of high frequency identifiers with a very small memory w.r.t. the length m of the input stream (the Count-Min data structure $\hat{F}$ is a 15×14 array) .

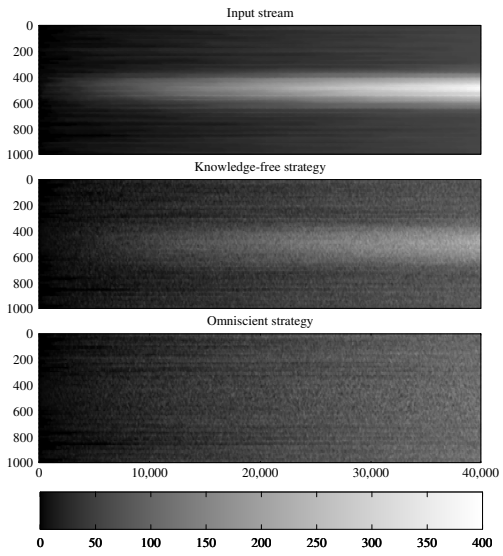

FIGURE 1: Frequency distribution as a function of time. Settings : $m = 40,000$, $n = 1000$, $c = 15$, $k = 15$, $s = 14$.

Références

[ABS13] E. Anceaume, Y. Busnel, and B. Sericola. Uniform node sampling service robust against collusions of malicious nodes. In *the 43rd Intl Conf. on Dependable Systems and Networks (DSN 2013)*, 2013.

[BGK+09] E. Bortnikov, M. Gurevich, I. Keidar, G. Kliot, and A. Shraer. Brahms : Byzantine Resilient Random Membership Sampling. *Computer Networks*, 53 :2340–2359, 2009.

[CM05] G. Cormode and S. Muthukrishnan. An improved data stream summary : the count-min sketch and its applications. *Journal of Algorithms*, 55(1) :58–75, 2005.