J. Parallel Distrib. Comput. 77 (2015) 84–94

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Efficient random walk sampling in distributed networks[☆]

Atish Das Sarma [a], Anisur Rahaman Molla [b,*], Gopal Pandurangan [b,c]

[a] eBay Research Labs, eBay Inc., CA, USA
[b] Division of Mathematical Sciences, Nanyang Technological University, Singapore 637371, Singapore
[c] Department of Computer Science, Brown University, Providence, RI 02912, USA

HIGHLIGHTS

- We consider the problem of sampling nodes through random walks in a distributed network.
- We present algorithm to compute several random walk samples in a continuous online fashion.
- Our algorithm minimizes (almost optimal) the number of rounds and messages.
- Our algorithm is effective and efficient as shown by experimental evaluation on various topological networks.
- Our algorithm performs very well on various metrics for all parameter ranges.

ARTICLE INFO

Article history:
Received 3 May 2014
Received in revised form
2 October 2014
Accepted 2 January 2015
Available online 9 January 2015

Keywords:
Random walks
Random sampling
Decentralized computation
Distributed algorithms
Self-aware network

ABSTRACT

Performing random walks in networks is a fundamental primitive that has found numerous applications in communication networks such as token management, load balancing, network topology discovery and construction, search, and peer-to-peer membership management. While several such algorithms are ubiquitous, and use numerous random walk samples, the walks themselves have always been performed naively.

In this paper, we focus on the problem of performing random walk sampling efficiently in a distributed network. Given bandwidth constraints, the goal is to minimize the number of rounds and messages required to obtain several random walk samples in a continuous online fashion. We present the first round and message optimal distributed algorithms that present a significant improvement on all previous approaches. The theoretical analysis and comprehensive simulations of our algorithms show that they perform very well in different types of networks of differing topologies.

In particular, our results show how several random walks can be performed continuously (when source nodes are provided only at runtime, i.e., online), such that each walk of length ℓ can be performed exactly in just $\tilde{O}(\sqrt{\ell D})$ rounds[1] (where D is the diameter of the network), and $O(\ell)$ messages. This significantly improves upon both, the naive technique that requires $O(\ell)$ rounds and $O(\ell)$ messages, and the sophisticated algorithm of Das Sarma et al. (2013) that has the same round complexity as this paper but requires $\Omega(m\sqrt{\ell})$ messages (where m is the number of edges in the network). Our theoretical results are corroborated through extensive simulations on various topological data sets. Our algorithms are fully decentralized, lightweight, and easily implementable, and can serve as building blocks in the design of topologically-aware networks.

1. Introduction

Random walks play a central role in computer science, spanning a wide range of areas in both theory and practice, including distributed computing and communication networks. Algorithms in many different applications use random walks as an integral subroutine. Applications in communication networks include token management [10,17,31], load balancing [32], small-world routing [34], search [1,15,28,37,46], information propagation and

---

[☆] A short version of this paper appeared in the Proceedings of 31st Annual IEEE International Conference on Computer Communications (IEEE INFOCOM), 2012, pp. 2906–2910, [20].

[*] Corresponding author.

E-mail addresses: atish.dassarma@gmail.com (A. Das Sarma), anisurpm@gmail.com (A.R. Molla), gopalpandurangan@gmail.com (G. Pandurangan).

[1] Throughout this paper, $\tilde{O}$ hides polylogarithmic factors in the number of nodes in the network.

http://dx.doi.org/10.1016/j.jpdc.2015.01.002

gathering [12,33], network topology construction [28,35,36], checking expander [25], constructing random spanning trees [7,8,13], monitoring overlays [39], group communication in ad hoc network [24], gathering and dissemination of information over a network [2], distributed construction of expander networks [35], and peer-to-peer membership management [26,47]. Random walks have also been used to provide uniform and efficient solutions to distributed control of dynamic networks [14]. [46] describes a broad range of network applications that can benefit from random walks in dynamic and decentralized settings. For further references on applications of random walks to distributed computing and networks, see, e.g., [14,46].

A key purpose of random walks in network applications is to perform node sampling. Random walk-based sampling is simple, local, and robust. Random walks also require little index or state maintenance which make them especially attractive to self-organizing dynamic networks such as Internet overlay and ad hoc wireless networks [14,46]. In this paper we present efficient distributed random walk sampling algorithms in networks that are significantly faster than the existing and naive approaches and at the same time achieve optimal message complexity. Our simulation results further show that our techniques perform very well in various network topologies.

While the sampling requirements in different applications vary, whenever a true sample is required from a random walk of certain steps, all applications perform the walks naively—by simply passing a token from one node to its neighbor: thus performing a random walk of length ℓ takes time and messages that are linear with respect to ℓ. Such an algorithm may not scale well as the network size increases and hence it is better to investigate algorithms with sub-linear time and message complexity. Previous work in [23] shows how to (partially) overcome this hurdle through a quadratic improvement in time and perform random walks optimally, i.e. in $\tilde{O}(\sqrt{\ell D})$ rounds. However, their algorithm requires a large number of messages for every random walk, depending on the number of edges in the network. The algorithm presented here shows how to perform the walks with *optimal* message complexity, i.e. just $O(\ell)$ messages per walk amortized, without compromising at all on the worst case round complexity. Further, the previous paper [23] only considered performing a single walk, or a few walks. Most applications, however, require several walks to be performed in a continuous manner. This continuous processing of walks is of specific importance in distributed networks and our results are applicable in this general framework.

Our random walk sampling algorithm can be a useful building block in the design of *topologically (self-)aware* networks, i.e., networks that can monitor and regulate themselves in a decentralized fashion. For example, efficiently computing the mixing time or the spectral gap, allows the network to monitor connectivity and expansion properties of the network [23]. This paper is an enhanced version of the short paper [20], which appeared in IEEE INFOCOM 2012.

Our contributions

1. We introduce the problem of *continuous* processing of random walks. The objective is for a network to support a continuous sequence of random walk requests from various source nodes and perform node sampling to minimize round and message complexity for each request.
2. We present the first algorithm that is efficient in both round complexity and message complexity. Our technique and analysis present almost-tight bounds on the message and round complexity in a widely used network congestion model.
3. We perform comprehensive experimental evaluation on numerous topological networks and highlight the effectiveness and efficiency of our algorithm. The experimental results corroborate the theoretical contributions and show that our random walk sampling algorithm performs very well on various metrics for all parameter ranges.

Two key features that distinguish the present work from previous work on distributed random walks are as follows:

- *Continuous processing of walks*: Unlike previous works (e.g., [21–23,42,19]), which deal with one or multiple random walks in a "one-shot" manner, this paper addresses the problem of continuous random walk sampling requests. In many applications, random samples have to be computed continuously, and hence the current results are more suitable in that setting.
- *Efficient in both message and round complexity*: Previous work [22,23] focused on the goal of reducing the round complexity of distributed random walks. In particular, for performing a single random walk of length ℓ, an $\tilde{O}(\sqrt{\ell D})$ time algorithm was presented. However, the message complexity of this algorithm was $\tilde{O}(m\sqrt{\ell D} + n\sqrt{\ell/D})$; this is quite high compared to the optimal $O(\ell)$ message complexity. It is not clear as to how the above message complexity can be improved for a single random walk. However, as shown in the current paper, for the problem of continuous random walks, one can improve the *amortized* message complexity to $\tilde{O}(\ell)$, which is almost optimal.

Overview. In the remaining part of this section, we discuss the network model, related work, and the formal notation and problem formulation considered in this paper. We present our algorithms, and message and round complexity analyses in Section 2. This rests on some concentration analysis of key random walk properties of our algorithm that we then prove in Section 3. Finally, we present extensive simulations on various topological networks in Section 4.

1.1. Distributed network model

We model the communication network as an undirected, unweighted, connected n-node graph $G = (V, E)$. Every node has limited initial knowledge. Specifically, assume that each node is associated with a distinct identity number (e.g., its IP address). At the beginning of the computation, each node v accepts as input its own identity number and the identity numbers of its neighbors in G. The node may also accept some additional inputs as specified by the problem at hand. The nodes are allowed to communicate through the edges of the graph G. We assume that the communication occurs in synchronous *rounds*. We will use only small-sized messages. In particular, in each round, each node v is allowed to send a message of size $O(\log n)$ through each edge $e = (v, u)$ that is adjacent to v. The message will arrive to u at the end of the current round. This is a widely used standard model to study distributed algorithms (e.g., see [43,44]) and captures the bandwidth constraints inherent in real-world computer networks. Our algorithms can be easily generalized if B bits are allowed (for any pre-specified parameter B) to be sent through each edge in a round. Typically, as assumed here, $B = O(\log n)$, which is number of bits needed to send a node ID in an n-node network.

While this is a nice theoretical abstraction, it still does not motivate the most natural practical difficulties. A well established concern with this model is that for simple operations, the entire network may spawn a large number of parallel messages in order to minimize rounds. This can be very expensive from a practical standpoint. A critical component in the analysis of practical algorithms is the overall message complexity per execution of any algorithm. This becomes even more crucial from the standpoint