Brief Announcement: Ultra-Fast Asynchronous Randomized Rumor Spreading

Ali Pourmiri
Macquarie University
Sydney, Australia
alipourmiri@gmail.com

Fahimeh Ramezani
Dept. of Mathematics, University of Isfahan
Isfahan, Iran
f.ramezani@sc.ui.ac.ir

ABSTRACT

Standard randomized rumor spreading algorithms propagate a piece of information, so-called the rumor, in a given network that proceed in synchronized rounds. Starting with a single informed node, in each subsequent round, every node calls a random neighbor in order to exchange the rumor (by sending the rumor to the neighbor (push algorithm) or asking it from the neighbor (pull algorithm)). Panagiotou et al. [ISAAC'13] considered a multiple-call version of the algorithms where each node is enabled to make more than one call in each round. The number of calls of a node is independently chosen from a probability distribution R. Seeking for a more realistic model, we propose an asynchronous version of the multiple-call algorithms on fully connected networks. In our model, each node has an independent Poisson clock whose rate may differ from others. Basically, the clock rate of each node is independently drawn from a probability distribution R at the beginning of the process. The push algorithm starts with a single informed node, when the clock of an informed node rings, the node contacts a random neighbor and sends (pushes) the rumor to the neighbor. Similarly, in the push-pull, if the clock of a node rings, then the node contacts a random neighbor in order to exchange the rumor. We study the effect of R on the *spreading time* of the algorithms, which is the time that the algorithm needs to inform all nodes. In this work, we show that if R is a power law distribution with exponent $\beta \in (2, 3)$ and $\varepsilon \in [1/n, 1 - 1/n]$ be an arbitrary number. Then, in expectation, after $\mathcal{O}(1+\log(1/\varepsilon))$ time the push-pull algorithm informs at least $(1 - \varepsilon)n$ nodes. Moreover, if R is an arbitrary distribution with bounded mean and variance, we show that the push algorithm spreads the rumor in a complete network with n nodes in $\frac{2\log n}{\mathbf{E}[R]} \pm O(\log\log n)$ time, with high probability.

CCS CONCEPTS

• **Mathematics of computing** → **Probabilistic algorithms**; • **Theory of computation** → **Network flows**.

SPAA '19, June 22–24, 2019, Phoenix, AZ, USA

ACM ISBN 978-1-4503-6184-2/19/06.
https://doi.org/10.1145/3323165.3323167

KEYWORDS

randomized rumor spreading, push/pull, asynchronous models

ACM Reference Format:
Ali Pourmiri and Fahimeh Ramezani. 2019. Brief Announcement: Ultra-Fast Asynchronous Randomized Rumor Spreading. In *31st ACM Symposium on Parallelism in Algorithms and Architectures (SPAA '19), June 22–24, 2019, Phoenix, AZ, USA*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3323165.3323167

1 INTRODUCTION

Standard randomized rumor spreading algorithms are important primitives for disseminating a piece of information, so-called the rumor, in large and complex networks. One basic variant of these algorithms is the *standard push* algorithm which proceeds in synchronized rounds. Initially, an arbitrary node in a given network knows the rumor. Then, in each subsequent round, every informed node selects a neighbor uniformly at random and sends (pushes) the rumor to that neighbor. Similarly, in the *standard pull* algorithm, in each round, every uninformed node calls a random neighbor and tries to learn the rumor from it. Moreover, in the *standard push-pull* algorithm, every node contacts a random neighbor and they learn the rumor from each other, if at least one of them knows it. The algorithms are based on a simple idea that, in each round, every node may contact a random neighbor which causes them to be local, scalable and robust against network failures (cf. [9, 11]). Demers et al. [6] first introduced the standard push-pull protocol to propagate an update in a network that consists of replicated databases. Subsequently, the rumor spreading algorithms have been successfully applied in many settings such as failure detection [25], resource discovery [18], load balancing [4], data aggregation [21], and analysis of the spread of computer viruses [3]. A well-studied parameter related to the algorithms (i.e., push, pull, and push-pull) on a network is the *spreading time* which is the time for which all nodes of the network become informed. Doerr et al. [7] proposed an algorithm in order to reduce the *message complexity*, which is the total number of messages sent by entities of the network during the algorithm execution. Panagiotou et al. [22] considered a variation of the rumor spreading algorithms, so-called the multiple-call protocols, where each node is enabled to contact more than one random neighbor in each round. More precisely, before the algorithm starts, each node, say u, picks a random number r_u from a given probability distribution R over positive integers. Starting with a single informed node, in each round, every node u calls r_u random neighbor(s) in order to exchange the rumor (using push, pull, or push-pull operation). In this model, provided R has a bounded mean, the message complexity increases by at most a constant factor while the rumor spreads faster than the standard randomized rumor spreading algorithms.

Besides the synchronized rumor spreading algorithms, an *asynchronous* version of the algorithms was defined so that nodes do not act in a synchronized manner, instead each node has a clock that ticks according to the arrival times of a rate 1 Poisson process. When the clock of a node ticks, the node contacts a randomly selected neighbor to exchange the rumor (i.e., either push the rumor to the neighbor or pulls the rumor from the neighbor). Motivated by applications in sensor, peer-to-peer, social, and ad hoc networks where a centralized clock does not exist, Boyd et al. [4] considered the asynchronous push-pull algorithm. In a different context, Janson [19] showed the asynchronous push-pull requires $\log n + \mathcal{O}(1)$ time to spread the rumor in a complete network on n nodes.

In a large and complex network, entities may have different communication power and act according to their personalized clocks. Seeking for a more realistic model, we consider an asynchronous version of the multiple-call protocols, where each node, say u, has an independent Poisson clock with rate r_u, where r_u is a random number drawn from R. We sometimes refer to the model as multiple-rate algorithms.

1.1 Our results

In this paper, we throughly study a variation of the asynchronous push and push-pull protocols on fully connected networks where the clock of each node ticks at arrival times of a Poisson process whose rate may differ from the others. More precisely, let $[n] = \{1, 2, \ldots, n\}$ be set nodes of the network and R denotes a given probability distribution over $[1, \infty)$. We will assume that every node $i \in [n]$ has an independent Poisson clock with rate r_i where each r_i, $i \in [n]$, is independently drawn from distribution R, at the beginning of the algorithm. Starting with a single node which knows the rumor, the push algorithm proceeds in asynchronous rounds. When the clock of an informed node ticks, then the node calls a random neighbor and transmits the rumor to it. We let T_{ap} be the random time that the push algorithm requires to propagate the rumor to all n nodes of the network. The push-pull algorithm is also defined so that when the clock of a node rings, then the node contacts a randomly selected neighbor and they exchange the rumor if at least one of them is aware of the rumor. For every $\varepsilon \in [1/n, 1-1/n]$, we will use $T_{app,\varepsilon}$ to denote the random time that the push-pull algorithm needs to inform $(1-\varepsilon)n$ nodes. Our first result concerns the push algorithm for which we assume that R has a bounded mean and variance. Here, we show that the spreading time of the protocol is concentrated around its mean.

Theorem 1.1. *Suppose that R is a given distribution with mean $\mu = \mathcal{O}(1)$ and variance $\sigma^2 = O(1)$. Then,*

$$\mathbf{E}[T_{ap}] = \frac{2\log n}{\mu} \pm O(1).$$

Moreover, with probability $1 - o(1)$, $T_{ap} = \frac{2\log n}{\mu} \pm \omega(\sqrt{\log n})$.

In the next result, we analyze the push-pull protocol for which probability distribution R follows a power law with $\beta \in (2, 3)$. We consider the power law distributions as they have been observed in many natural phenomena such as degree distribution of complex networks [2], file popularities in cache networks [5] and etc. More specifically, the power law distribution with $\beta \in (2, 3)$ has a bounded mean and unbounded variance. Our proof technique for showing this result might be of independent interest.

Theorem 1.2. *Suppose that R is a power law distribution with $\beta \in (2, 3)$ and we have a fully connected network of size n. Before the algorithm starts each node u has a Poisson rate r_u randomly drawn from distribution R. Let $\varepsilon \in [1/n, 1-1/n]$ be an arbitrary number. Then, we have*

$$\mathbf{E}[T_{app,\varepsilon}] = \mathcal{O}(1 + \log 1/\varepsilon).$$

To show the expected spreading time, we analyze the algorithm in three consecutive phases namely, initial, middle and final phase. In the initial phase, we show that the protocol, in expectation, requires $\mathcal{O}(1)$ time to inform $(n/\log^2 n)^{1/\beta-1}$ nodes. To do so, we define a sequence of deterministic integers, namely, $w_0, w_1, \ldots$, where w_0 is constant and show that there exists $k = O(\log\log n)$ such that $w_{2k} \geqslant (n/\log^2 n)^{1/\beta-1}$. Here, we alternatively consider the pull and push operation. For some $i \geqslant 0$, let us start with w_{2i} informed nodes. Considering only the pull operation, a node u with $r_u \geqslant w_{2i+1}$ will be informed in 2^{-i} time, in expectation. When u gets informed, we consider the push algorithm and inform w_{2i+2} nodes in 2^{-i} time and hence the algorithm informs w_{2k}, in $\sum_i 2^{-i+1} = \mathcal{O}(1)$ time. The middle phase starts with $(n/\log^2 n)^{1/\beta-1}$ informed nodes and ends with $\Omega(n)$ informed nodes. In this phase, however, we define a different sequences, we apply a similar technique to show the average time is a constant. The final phase starts with at least n/c informed nodes for some constant c and ends when $(1-\varepsilon)n$ nodes get informed. Recall that the clock of every uniformed node ticks at arrival times of a Poisson process of rate at least 1. Each pull attempt calls an informed node with probability at least $1/c$, because n/c nodes were informed in the previous phase. Therefore, the waiting time for an uninformed node to get informed is distributed as an exponential distribution with constant rate. Applying the order statistics of exponential random variables shows that the phase ends after $\mathcal{O}(\log 1/\varepsilon)$ time, in expectation.

1.2 Related Work

Researchers have extensively studied the spreading time of the synchronous push and push-pull protocols. In one of the first papers in this area, Fireze and Grimmett [14] analyzed the spreading time of the synchronous push protocol on a fully connected network with n nodes and showed that the spreading time is $\log n \pm o(\log n)$. The result was later strengthened by Pittel [24]. Karp et al. [20] studied the spreading time and the message complexity of the synchronous push-pull on fully connected networks. They showed that using $\mathcal{O}(n \log\log n)$ messages the protocol requires $\log_3 n + \mathcal{O}(\log\log n)$ rounds to inform all n nodes. Besides the complete network, the protocols have been studied on various network topologies (e.g., see [10–12]). Giakkoupis [15, 16] derived an upper bound for the spreading time of the synchronous push-pull algorithm in terms of expansion profile of the network that is $\mathcal{O}(\min\{\frac{\log n}{\Phi}, \frac{\log n \log \Delta}{\alpha}\})$, where Φ , α, and Δ denote the conductance, vertex expansion, and maximum degree of the network, respectively.Seeking for a more realistic model, recently, the asynchronous rumor spreading protocols on networks have also received much attention. Boyd et al. [4] proposed the asynchronous push-pull protocol on the complete network in order to drop the assumption that the nodes act in a synchronized manner. In this model, each node has an independent Poisson clock with rate 1. When the clock of a node rings, the node contacts a random neighbor to exchange the rumor. Panagiotou and Speidel [23] studied the spreading time of the asynchronous push-pull protocol on Erdös-Rényi random graphs $G_{n,p}$, for any $p > \log n/n$. They showed that the spreading time is $\log n + \mathcal{O}(1)$ which is almost unaffected by p. Moreover, they quantified the robustness of the protocol with respect to transmission and node failures. The algorithm was also studied on preferential attachments [8] and Chug-Lu random graphs [13]. Let G be a given network with n nodes and assume that $T_s(G)$ and $T_a(G)$ are the spreading time of a synchronous and asynchronous rumor spreading algorithm (push, pull or push-pull) on G, respectively. Acan et al. [1] studied the ratio $\frac{T_s(G)}{T_a G}$ and prove that $\Omega(1/\log n) \leqslant \frac{T_s(G)}{T_a G} \leqslant \mathcal{O}(n^{2/3})$. More recently, Giakkoupis et al. [17] also showed that $T_a(G) = \mathcal{O}(T_s(G) + \log n)$. Moreover, they improved the upper bound for $\frac{T_s(G)}{T_a G}$ to $n^{1/2}(\log n)^{\mathcal{O}(1)}$, settling down a conjecture in [1], positively.

REFERENCES

[1] Hüseyin Acan, Andrea Collevecchio, Abbas Mehrabian, and Nick Wormald. 2015. On the Push&Pull Protocol for Rumour Spreading: [Extended Abstract]. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing, PODC 2015, Donostia-San Sebastián, Spain, July 21 - 23, 2015*. 405–412. https://doi.org/10.1145/2767386.2767416

[2] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *Science* 286(5439) (1999), 509–512.

[3] Noam Berger, Christian Borgs, Jennifer T. Chayes, and Amin Saberi. 2005. On the S pread of V iruses on the I nternet. In *Proc. 16th Symp. Discrete Algorithms (SODA)*. 301–310.

[4] Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. 2006. Randomized gossip algorithms. *IEEE Transactions on Information Theory* 52, 6 (2006), 2508–2530.

[5] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker. 1999. Web Caching and Zipf-like Distributions: Evidence and Implications. In *Proceedings IEEE INFOCOM '99, The Conference on Computer Communications, Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, The Future Is Now, New York, NY, USA, March 21-25, 1999*. 126–134. https://doi.org/10.1109/INFCOM.1999.749260

[6] Alan Demers, Mark Gealy, Dan Greene, Carl Hauser, Wes Irish, John Larson, Sue Manning, Scott Shenker, Howard Sturgis, Dan Swinehart, Doug Terry, and Don Woods. 1987. Epidemic algorithms for replicated database maintenance. In *Proc. 6th Symp. Principles of Distributed Computing (PODC)*. 1–12.

[7] Benjamin Doerr and Mahmoud Fouz. 2011. Asymptotically Optimal Randomized Rumor Spreading. In *Proc. 38th Intl. Coll. Automata, Languages and Programming (ICALP)*. 502–513.

[8] Benjamin Doerr, Mahmoud Fouz, and Tobias Friedrich. 2012. Asynchronous Rumor Spreading in Preferential Attachment Graphs. In *Proc. 13th Scandinavian Workshop Algorithm Theory (SWAT)*. 307–315.

[9] Robert Elsässer and Thomas Sauerwald. 2006. On the Runtime and Robustness of Randomized Broadcasting. In *Algorithms and Computation, 17th International Symposium, ISAAC 2006, Kolkata, India, December 18-20, 2006, Proceedings*. 349–358. https://doi.org/10.1007/11940128_36

[10] Robert Elsässer and Thomas Sauerwald. 2008. The power of memory in randomized broadcasting. In *Proc. 19th Symp. Discrete Algorithms (SODA)*. 218–227.

[11] Uriel Feige, David Peleg, Prabhakar Raghavan, and Eli Upfal. 1990. Randomized Broadcast in Networks. *Random Struct. Algorithms* 1, 4 (1990), 447–460.

[12] Nikolaos Fountoulakis and Konstantinos Panagiotou. 2010. Rumor Spreading on Random Regular Graphs and Expanders. In *Proc. 14th Intl. Workshop on Randomization and Comput. (RANDOM)*. 560–573.

[13] Nikolaos Fountoulakis, Konstantinos Panagiotou, and Thomas Sauerwald. 2012. Ultra-fast rumor spreading in social networks. In *Proc. 23th Symp. Discrete Algorithms (SODA)*. 1642–1660.

[14] Alain Frieze and Geoffrey Grimmett. 1985. The shortest-path problem for graphs with random-arc-lengths. *Discrete Applied Mathematics* 10 (1985), 57–77.

[15] George Giakkoupis. 2011. Tight bounds for rumor spreading in graphs of a given conductance. In *Proc. 28th Symp. Theoretical Aspects of Computer Science (STACS)*. 57–68.

[16] George Giakkoupis. 2014. Tight Bounds for Rumor Spreading with Vertex Expansion. In *Proc. 25th Symp. Discrete Algorithms (SODA)*. 801–815. https://doi.org/10.1137/1.9781611973402.59 arXiv:http://epubs.siam.org/doi/pdf/10.1137/1.9781611973402.59

[17] George Giakkoupis, Yasamin Nazari, and Philipp Woelfel. 2016. How Asynchrony Affects Rumor Spreading Time. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing, PODC 2016, Chicago, IL, USA, July 25-28, 2016*. 185–194. https://doi.org/10.1145/2933057.2933117

[18] Mor Harchol-Balter, Frank Thomson Leighton, and Daniel Lewin. 1999. Resource Discovery in Distributed Networks. In *Proc. 18th Symp. Principles of Distributed Computing (PODC)*. 229–237.

[19] Svante Janson. 1999. One, Two And Three Times Log N/N For Paths In A Complete Graph With Random Weights. *Combinatorics, Probability & Computing* 8, 4 (1999), 347–361. http://journals.cambridge.org/action/displayAbstract?aid=46717

[20] Richard M. Karp, Christian Schindelhauer, Scott Shenker, and Berthold Vöcking. 2000. Randomized Rumor Spreading. In *Proc. 41st Symp. Foundations of Computer Science (FOCS)*. 565–574.

[21] David Kempe, Alin Dobra, and Johannes Gehrke. 2003. Gossip-Based Computation of Aggregate Information. In *Proc. 44th Symp. Foundations of Computer Science (FOCS)*. 482–491.

[22] Konstantinos Panagiotou, Ali Pourmiri, and Thomas Sauerwald. 2013. Faster Rumor Spreading with Multiple Calls. In *Proc. 24th Intl. Symp. Algorithms and Computation (ISAAC)*. 446–456.

[23] Konstantinos Panagiotou and Leo Speidel. 2013. Asynchronous Rumor Spreading on Random Graphs. In *Algorithms and Computation - 24th International Symposium, ISAAC 2013, Hong Kong, China, December 16-18, 2013, Proceedings*. 424–434. https://doi.org/10.1007/978-3-642-45030-3_40

[24] Boris Pittel. 1987. On spreading a rumor. *SIAM J. Appl. Math.* 47, 1 (1987), 213–223.

[25] Robbert van Renesse, Yaron Minsky, and Mark Hayden. 1998. A Gossip-Style Failure Detection Service. In *Proceedings of the 15th IFIP International Conference on Distributed Systems Platforms (Middleware'98)*. 55–70.