Brief Announcement: Optimal Address-Oblivious Epidemic Dissemination

Hugues Mercier
Schools of Computer Science and Mathematics
Université de Neuchâtel
Neuchâtel, Switzerland
hugues.mercier@unine.ch

Laurent Hayez
Schools of Computer Science and Mathematics
Université de Neuchâtel
Neuchâtel, Switzerland
laurent.hayez@unine.ch

Miguel Matos
Instituto Superior Técnico
Universidade de Lisboa
INESC-ID
Lisboa, Portugal
miguel.matos@inesc-id.pt

ABSTRACT

We consider the problem of reliable gossip/epidemic dissemination in a network of n processes using push and pull algorithms. We generalize the random phone call model so that processes can refuse to push a rumor or answer pull requests. With this relaxation, we show that it is possible to disseminate a rumor to all processes with high probability using $\Theta(\ln n)$ rounds of communication and only $n + O(\frac{n}{\ln n})$ messages, both of which are optimal and achievable with push-pull and pull-only algorithms. Our algorithms are strikingly simple, address-oblivious and thus fully distributed. This contradicts a well-known result of Karp et al. [3] stating that any address-oblivious algorithm requires $\Omega(n \ln \ln n)$ messages.

We also develop precise estimates of the number of rounds required in the push and pull phases of our algorithms to guarantee dissemination to all processes with a certain probability. For the push phase, we focus on a practical infect upon contagion approach that balances the load evenly across all processes. As an example, our push-pull algorithm requires 17 rounds to disseminate a rumor to all processes with probability $1 - 10^{-100}$ in a network of one million processes with a communication overhead of only 0.4%.

1 INTRODUCTION

Since the proposal to update replicated databases [2], gossip algorithms have been used to address a wide variety of problems. The randomness inherent to the selection of the communication peers makes gossip algorithms particularly robust to all kinds of failures such as message loss and process failures, which tend to be the norm rather than the exception in large systems. Their appeal also stems from their simplicity and highly distributed nature.

Push algorithms. The simplest epidemic dissemination algorithms are push-based, where processes that know the rumor propagate it to other processes. Consider the following "infect forever" push algorithm. The algorithm starts with a single process knowing a rumor, and at every round, every informed process pushes the rumor to a process chosen uniformly and independently at random. There are other flavors of push algorithms such as an "infect upon contagion" approach [4] where a process propagates the rumor immediately after it is received regardless of previous receptions. Unfortunately, push algorithms must transmit $\Theta(n \ln n)$ messages if every process is to learn a rumor with high probability[1].

Pull algorithms. Instead of pushing a rumor, a different strategy is for an uninformed process to ask an interlocutor chosen at random to convey the rumor if it is already in its possession. Pulling algorithms are advantageous when rumors are frequently created because pull requests will more often than not reach processes with new rumors to share. However, pull requests in systems without activity result in useless traffic.

Push-pull algorithms and the random phone call model. The idea to push and pull rumors simultaneously was first considered by Demers et al. [2], and further studied by Karp et al. [3] who considered the following random phone call model. At each round, each process randomly chooses an interlocutor and calls it. If, say, Alice calls Bob, once the communication is established, Alice pushes the rumor to Bob if she has it, and pulls the rumor from Bob if he has it. Establishing communication (the phone call itself) is free, and only messages that include the rumor are counted. Using the random phone call model, Karp et al. [3] presented a push-pull algorithm that transmits a rumor to every process with high probability using $O(\ln n)$ rounds of communication and $O(n \ln \ln n)$ messages.

1.1 Our contributions

Generalized random phone call model. In the original random phone call model, rumors are transmitted in both directions whenever both players on the line have the rumor. We remove this restriction and generalize the model: at each communication round, each process calls between 0 and f_{in} processes uniformly at random to request a rumor, calls between 0 and f_{out} processes uniformly at random to push a rumor, and has the option not to answer pull requests. We assume, like for the original model, that establishing the communication is free, and we only count the number of messages that contain the rumor. We also assume that there is a counter attached to the rumor keeping track of the number of dissemination rounds since its creation, that processes can reply to pull requests in the same round, that the rounds are synchronous, and that processes have a complete view of the system.

Breaking the lower bounds from [3]. In their seminal work, the authors define the model such that processes do not have to share the rumor once the communication is established. They state

[1]With high probability (w.h.p) means with probability at least $1 - O(n^{-c})$ for a constant $c > 0$.

PODC'17, July 25-27, 2017, Washington, DC, USA
 978-1-4503-4992-5/17/07…$15.00
DOI: http://dx.doi.org/10.1145/3087801.3087862

that any algorithm in the random phone call model running in $O(\ln n)$ rounds with communication peers chosen uniformly at random requires $\omega(n)$ messages, and that any address-oblivious algorithm needs $\Omega(n \ln \ln n)$ messages to disseminate a rumor. Address-obliviousness means that the decisions taken locally by each process are oblivious to the address of the other processes. However, in the proofs of their lower bounds it is implicitly assumed that processes always pull and push the rumor once the connection is established, which, it turns out, is not optimal and allows us to break both lower bounds.

Optimal algorithms with $O(\ln n)$ rounds and $O(n)$ messages. If we discount the cost of establishing the communication (the phone call), it is natural to let processes choose whether or not to call, and whether or not to reply when called. This generalization makes a huge difference: we present a pull-only algorithm and a push-pull algorithm that disseminate a rumor to all processes with high probability in $O(\ln n)$ rounds of communication using only $O(n)$ messages. The idea is simple: we do not push old rumors because doing so results in a large communication overhead.

Our pull-only algorithm obviously never pushes rumors: uninformed processes attempt to pull the rumor from f_{in} processes per round until they receive it, and informed processes always reply to pull requests. Our infect-forever-push-pull algorithm first pushes the rumor during r_{push} rounds, stopping when at least $\frac{n}{\ln n}$ processes are informed with high probability. During this push phase, informed processes push the rumor to f_{out} processes at every round in an infect forever fashion. Uninformed processes send pull requests during the push phase, but informed processes do not reply to them. After r_{push} rounds, informed processes stop pushing the rumor and a pull-only phase is executed for r_{pull} rounds until all processes have received the rumor with high probability. Informed processes always reply to pull requests during this pull phase.

Both our pull and push-pull algorithms are optimal for the generalized phone call model. Their message complexity is optimal, their bit complexity is optimal, and we prove that their round complexity is also optimal by showing that pushing and pulling at the same time using potentially complex rules is not necessary: any algorithm in the generalized random phone call model requires $\Omega(\log_{\max(f_{\text{in}}, f_{\text{out}})+1} n)$ rounds of communication to disseminate a rumor with high probability.

Toward practical implementations. We derive precise analytic and numerical bounds on the number of rounds required for the push and pull phases of our algorithms. For the push phase, we focus on the infect upon contagion model for which, in round r, processes transmit all the rumors received in round $r-1$ even when they are not received for the first time. This model is of particular interest to us because it spreads the load uniformly across all processes as opposed to infect forever algorithms, while having the same asymptotic behavior. It is thus well-suited for practical implementations, especially if the payload (rumor) is large. Our infect-upon-contagion-push-pull algorithm shares the optimal properties of the infect-forever-push-pull algorithm. Furthermore, by adjusting the number of messages transmitted during the last round of the push phase, we can decrease the number of messages to $n + O(\frac{n}{\ln n})$ even when $f_{\text{out}} \in O(\ln n)$.

2 ANALYSIS OF THE INFECT UPON CONTAGION PUSH MODEL

We model the infect upon contagion push model as an occupancy problem where messages are abstracted as balls, and processes play the role of bins. Let X_r be the number of bins that receive at least one ball during round r. We show that $\mathbb{E}[X_{r+1}] = \mathbb{E}[\varphi(X_r)] \leq \psi(r+1)$, where $\psi(r+1) \triangleq \varphi(\psi(r)) = n\left(1 - \left(1 - \frac{1}{n}\right)^{f_{\text{out}} \cdot \psi(r)}\right)$ and $\varphi(x) \triangleq n\left(1 - \left(1 - \frac{1}{n}\right)^{f_{\text{out}} \cdot x}\right)$. We lower bound the function $\psi(r)$ by a logistic equation. It is easy to see that $\psi(r)$ converges to a limit γ because it is monotonically increasing and bounded above by n. A solution for γ is given by Corless et al. [1] with the principal branch of the Lambert-W function: $\gamma = n\frac{f_{\text{out}} + W(-f_{\text{out}} e^{-f_{\text{out}}})}{f_{\text{out}}}$.

Lemma 2.1. *Let $X(r) = \frac{\gamma f_{\text{out}}^r}{\gamma + f_{\text{out}}^r - 1}$. If $f_{out} \geq 2$, then $\psi(r) \geq X(r)$ for every $r \in \mathbb{N}$.*

We observed that the logistic equation $X(r)$ closely approximates $\psi(r)$ and the estimation $\hat{\mathbb{E}}[X_r]$ of $\mathbb{E}[X_r]$ obtained with numerical simulations, but it is not a true lower bound for $\mathbb{E}[X_r]$ because $\mathbb{E}[X_r]$ is slightly lower than $\psi(r)$. However, $\psi(r)$ and $\hat{\mathbb{E}}[X_r]$ are always almost superposed and we conjecture that $\psi(r) - \mathbb{E}[X_r] \leq 1$ when $n \geq 3$ and $f_{\text{out}} \geq 2$. This is more than sufficient to use $X(r)$ to accurately bound the number of rounds of the push phase of our algorithms when implemented in large-scale systems. This is a significant improvement over the bounds of [4], which are only discussed asymptotically for $f_{\text{out}} \in \Omega\left(\frac{\ln n}{\ln \ln n}\right)$. For instance, for $n = 10^4$ and $p_e = 10^{-15}$, the bounds from [4] require $f_{\text{out}} = 23$ and $r = 50$ (i.e., ≈ 10 million balls), whereas our bounds require $r = 41$ with $f_{\text{out}} = 2$, or $r = 10$ with $f_{\text{out}} = \lfloor \ln n \rfloor = 14$ (i.e., ≈ 500000 balls).

3 PULL ALGORITHMS

Besides the seminal work of Demers et al. [2] and Karp et al. [3], there are few results on pull-only algorithms. Our first conclusion is that pulling and pushing have the same asymptotic round complexity, which is implicit in [3]. On expectation, pulling is always at least as good as pushing, although the higher variance of pull at the early stage of the dissemination makes pulling less efficient when the rumor is new. However, the behavior reverses and then more: pulling is much more efficient than pushing when the rumor is old. The second albeit trivial conclusion is that $O(n)$ messages are sufficient to disseminate a rumor with high probability when the fan-in is constant.

Let u_r be the number of uninformed processes after round r. We show that if we start the pull phase with $u_0 = n - \frac{n}{\ln n}$ uninformed processes, then $\mathbb{E}[u_r] \geq n \cdot \left(1 - \frac{1}{\ln n}\right)^{(f_{\text{in}}+1)^r}$, and $\mathbb{E}[u_r] \geq n^{-c}$ if $r \leq \log_{f_{\text{in}}+1}(c+1) + \log_{f_{\text{in}}+1} \ln n - \log_{f_{\text{in}}+1} \ln \frac{\ln n}{\ln n - 1}$. The distribution of u_r is very closely concentrated around the mean, thus we can derive precise estimates of the number of pull rounds for practitioners. This is useful for the pull phase of our push-pull algorithms.

Lemma 3.1. *Let u_r be the number of informed processes at round r, let $1 \leq f_{out} = f_{in} \leq n-1$, and let $n \geq 4$.*

$$\mathbb{E}_{pull}[u_{r+1} | u_r] \leq \mathbb{E}_{push}[u_{r+1} | u_r] \quad (1)$$

where $\mathbb{E}_{pull}[u_{r+1}|u_r]$ and $\mathbb{E}_{push}[u_{r+1}|u_r]$ are the expected values of the number of uninformed processes at round $r+1$ given u_r uninformed processes at round r for the pull and push version, respectively.

LEMMA 3.2. *The number of pull rounds required to inform all processes with high probability starting from $\frac{n}{\ln n}$ informed processes is in $\Theta(\log_{f_{in}+1} \ln n)$.*

THEOREM 3.3. *Our pull-only algorithm disseminates a rumor to all processes with high probability in $\Theta(\log_{f_{in}+1} n)$ rounds of communication, which is optimal for the generalized random phone call model.*

THEOREM 3.4. *If $f_{in} \in O(1)$, then the total number of messages (replies to pull requests) required by the pull algorithm is in $\Theta(n)$.*

4 PUSH-PULL ALGORITHMS

We now discuss our push-pull algorithms. We push when the rumor is young, then pull when the rumor is old. Processes do not reply to pull requests if the rumor is new, nor do they push the rumor if it is old.

LEMMA 4.1. *$\Theta(\frac{n}{\ln n})$ messages transmitted uniformly at random are necessary and sufficient to inform at least $\frac{n}{\ln n}$ processes with high probability. More precisely, if $m \geq \frac{n}{\ln n} + \frac{n}{\ln^2 n}$ messages are transmitted uniformly at random, then the probability of informing less than $\frac{n}{\ln n}$ processes is at most*

$$2e^{-\frac{\left(1-\frac{1}{\ln n}-\left(1-\frac{1}{n}\right)^m\right)^2\left(n-\frac{1}{2}\right)}{1-\left(1-\frac{1}{n}\right)^{2m}}}. \qquad (2)$$

THEOREM 4.2. *If $f_{in} \in O(1)$ and $f_{out} \in O(\ln n)$, our push-pull algorithm can disseminate a rumor to all processes with high probability using $\Theta(n)$ messages. The algorithm requires $\log_{f_{out}+1} n + O(\ln \ln n)$ rounds using an infect forever push phase, and $\log_{f_{out}} n + O(\ln \ln n)$ rounds with the infect upon contagion push strategy when $f_{out} \geq 2$. If $f_{out} \in O(1)$ and $f_{in} = 1$, both algorithms require $n + O\left(\frac{n}{\ln n}\right)$ messages.*

THEOREM 4.3. *Let $f = \max(f_{in}, f_{out})$. Any protocol in the generalized random phone call model requires $\Omega(\log_{f+1} n)$ rounds of communication to disseminate a rumor to all processes with high probability.*

We now illustrate the performance of our infect-upon-contagion-push-pull algorithm. First, using the tools of Section 2 and Lemma 4.1, we calculate the number of push rounds required to inform at least $\frac{n}{\ln n}$ processes with a chosen probability of imperfect dissemination. The threshold is very sharp for networks with a few hundred processes or more: there is a r' such that the probability of informing $\frac{n}{\ln n}$ processes goes from ≈ 0 in round $r'-1$ to ≈ 1 in round r'. In practice this phase transition is the sweet spot to switch from the push phase to the pull phase. We then calculate the number of rounds of the pull phase using the tools of Section 3.

The communication overhead is in $O\left(f_{\text{out}} \cdot \frac{n}{\ln n}\right)$, and the transition phase is such that with a logarithmic fan-out, we sometimes go from slightly less than $\frac{n}{\ln n}$ informed processes in round $r_{\text{push}}-1$ to almost n informed processes in round r_{push}. In these situations, we can decrease the number of overhead messages by transmitting each message in round r_{push} with a scaling probability p_{scale} such that the number of informed processes after round r_{push} is above but close to $\frac{n}{\ln n}$ with high probability. This ensures that the communication overhead is in $O\left(\frac{n}{\ln n}\right)$ even with a logarithmic fan-out. We can easily calculate p_{scale} with Eq. (2) and the machinery of Section 2.

Figure 1 shows the communication overhead for varying system sizes, f_{out} and f_{in} for a probability of error of $p_e = 10^{-15}$. The $f_{\text{out}} = \lfloor \ln n \rfloor$ $f_{\text{in}} = 1$ configuration is of particular interest in practice because it provides sublogarithmic latency with a negligible communication and process overhead. More precisely, for systems with between ten thousand and one million processes, the algorithm requires between 13 and 15 rounds, a fan-out of size 9 to 13, and when scaling the last push round the communication overhead varies between 1.2% for ten thousand processes and 0.3% for one million processes. The 10^{-15} probability of imperfect dissemination is on par with the failure probability of modern hardware. If this is still insufficient, we can decrease the probability of imperfect dissemination to 10^{-100} by reducing the scaling of the last push round, which slightly increases the communication overhead (to 2.6% for ten thousand processes and 0.4% for one million processes), and pulling for two to three additional rounds.

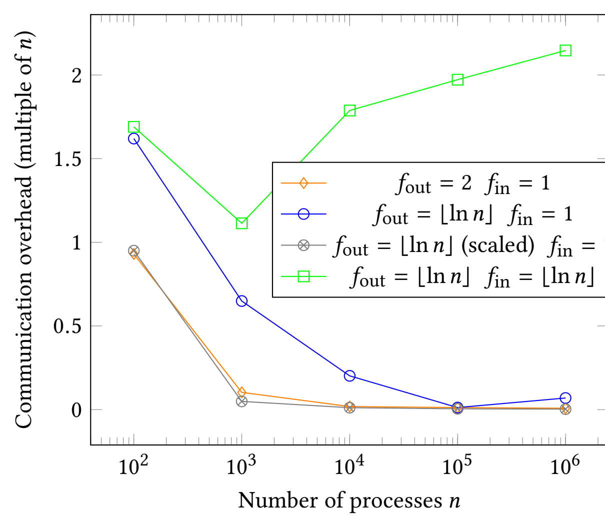

Figure 1: Communication overhead of the infect-upon-contagion-push-pull algorithm for $p_e = 10^{-15}$ and different f_{in}, f_{out} and network size n. Each data point is the average over 10 simulations. The simulations labeled (scaled) scale the last push round.

REFERENCES

[1] Robert M Corless, Gaston H Gonnet, David EG Hare, David J Jeffrey, and Donald E Knuth. 1996. On the Lambert *W* function. *Advances in Computational mathematics* 5, 1 (1996), 329–359.

[2] Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. 1987. Epidemic Algorithms for Replicated Database Maintenance. In *Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing (PODC '87)*. ACM, New York, NY, USA, 1–12. https://doi.org/10.1145/41840.41841

[3] Richard M. Karp, Christian Schindelhauer, Scott Shenker, and Berthold Vöcking. 2000. Randomized Rumor Spreading. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*. IEEE Computer Society, 565–574. https://doi.org/10.1109/SFCS.2000.892324

[4] Boris Koldehofe. 2004. Simple gossiping with balls and bins. *Stud. Inform. Univ.* 3, 1 (2004), 43–60.