Minimum-Cost First-Push-Then-Pull Gossip Algorithm

Ali Saidi
Advanced Infrastructure Design
Hamilton, NJ, USA
alx.saidi@gmail.com

Mojdeh Mohtashemi
MITRE Corp., McLean, VA, USA
MIT CSAIL, Cambridge, MA, USA
mojdeh@mitre.org

***Abstract*—In this paper, we study the communication overhead of gossip-based information dissemination algorithms. Among basic variants of gossip algorithm push is most efficient in the early rounds while, in contrast, pull becomes more efficient in the later rounds. Therefore, a cost-efficient gossip algorithm needs to combine the advantages of push and pull algorithms. One possible approach is to begin with push algorithm and then at some point switch to pull algorithm. We analyze the effect of transition round from push to pull on the communication cost of gossip algorithm. We use simple deterministic difference equations to approximately model the message propagation throughout the network for both push and pull algorithms and derive closed form solution for pull model. We then present our First-Push-Then-Pull (*FPTP*) gossip algorithm and obtain the optimum round to transition from push to pull. We show that, in a fully connected network, normalized communication cost is *minimized* to approximately a *constant* (≈2.6 transmissions/message/node) when the transition round is $\mathrm{Round}(\log N)$. Furthermore, we extend our results to networks with limited connectivity/cooperation and show that although the communication overhead increases moderately as a function of connection probability, it still remains approximately *constant*. To validate our results we test our algorithm in mobile ad-hoc network (MANET) environment using random-waypoint mobility model and show that the simulation results closely follow our analysis.**

Keywords- Gossip algorithms; Push; Pull; Mobile networks;

I. INTRODUCTION

Gossip (or epidemic) algorithms are a class of randomized communication protocol used for information dissemination over decentralized networks. They have shown to be an efficient, robust, and scalable technique to disseminate information in a wide range of distributed applications. Gossip algorithms have been proposed for many distributed applications, including database replication [1, 4], data aggregation [5], failure detection [6], system monitoring [7], broadcast and multicast [8-10], ad hoc network routing [11], and video streaming [12] among others. These randomized algorithms are locally very simple and have substantially higher degree of fault tolerance, compared to deterministic algorithms, which makes them highly scalable at the same time.

Gossip algorithms are based on pair wise random exchange of information where communication takes place in parallel rounds. In each round every node contacts another node at random and either receives or transmits a piece of information (or both). There are three basic variants of gossip algorithm -- Push, Pull, and Push&Pull:

- **Push:** in push algorithm each informed node randomly picks another node and sends its message.
- **Pull:** in pull algorithm each uninformed node randomly picks anther node and, if the selected node is informed, receives its message.
- **Push&pull:** in push&pull algorithm all the nodes pick another node at random and exchange their message if either node is informed.

Clearly, the advantages of randomized algorithms come at the price of increased communication cost and higher latency. Indeed communication overhead is substantial: $\Theta(N \ln N)$ transmissions per message for push algorithm, and $O(N \ln \ln N)$ for push&pull algorithm compared to $N-1$ transmission for deterministic algorithms. However it seems that the nonlinear relationship between cost and latency variables, in various randomized algorithms, introduces considerable difficulties in cost optimization.

Each one of the above randomized algorithms has its advantages and disadvantages in various stages of message propagation. Consider the communication cost of disseminating a message throughout the network. In the early rounds push algorithm is very efficient since every call from an informed node will reach an uninformed node with high probability. In pull algorithm, on the other hand, uninformed nodes place a lot of calls to other uninformed node and receive no updates. By contrast, in the later rounds of gossip algorithms pull becomes more and more efficient while push suffers from increasing number of informed nodes calling other informed nodes. Push&pull algorithm combines both the strength and weaknesses of the two algorithms that leads to faster convergence which comes at higher communication cost. In general, push algorithm could be less cost effective because it is possible for more than one informed nodes calling the same uninformed node.

The above argument leads us to conclude that a cost-efficient gossip algorithm will have to begin with push algorithm and then at some point switch to pull algorithm. The main problem then is how to determine when to switch from push to pull. In this paper we study the effect of transition round from push to pull algorithms on the communication cost of gossip algorithm. We use simple deterministic difference equations to model the message propagation throughout the network. These deterministic models are all based on the fact that for large networks the distribution of the proportion of the informed nodes in any given round is sharply concentrated around its mean value. In particular, Pittel [1] showed this property for push algorithm and Demers et al. [2] showed it for pull algorithm. Bakhshi et al. [13] used mean-field analysis to show the same property for push&pull algorithm. Most of the works on efficiency analysis of gossip algorithms however focus on deriving a bound on the number of rounds required to spread a message to the entire network and the communication cost is generally inferred from the number of rounds. In this paper we concentrate on minimizing the cost directly from the message propagation models and then analyze the resulting convergence time.

The remainder of the paper is organized as follows. In Section 2, we present an approximate information propagation model for pull algorithm using a deterministic difference equation along with its closed form solution and the associated communication cost function. In Section 3, we repeat the same analysis for push algorithm and compare the performances of the two algorithms. In Section 4, we present our *"First-push-then-pull"* algorithm and derive the optimum round to transition from push to pull algorithm to minimize the communication overhead. We then extend our model to limited connectivity/cooperation case in Section 5. In Section 6 we apply our algorithm to mobile ad-hoc network (MANET) and present simulation results for networks with limited connectivity and multiple message propagation cases. Finally, in Section 7 we draw conclusions and point to some further directions for research.

II. PULL DIFFUSION MODEL

We now study the difference equation approximating the message dissemination for pull algorithm in fully connected networks. Let X_i be the proportion of informed nodes in the network in round *i*. Then,

$$E(X_{i+1} \mid X_i) = X_i(1 - X_i) + X_i \quad \text{for } i \ge 0$$

Ignoring the fluctuation of X_{i+1} around its mean value, we have the following approximation for progression of the process X_i:

$$X_{i+1} = X_i(1 - X_i) + X_i \quad \text{for } i \ge 0 \tag{1}$$

The intuition behind (1) is that in round $i+1$, the probability that some node does not have the information (the proportion $1 - X_i$) contacts a node that does, is X_i . An Implicit assumption in (1) is that all the nodes cooperate in spreading the message. We can immediately infer some properties of (1). For example, it converges at the limit, that is, for $X_0 \in (0,\ 1)$ and $\varepsilon \in (0,\ 1 - X_0)$ there exists a τ such that $X_\tau = 1 - \varepsilon$. We say that τ is the number of rounds in which *X* converges.

To understand the behavior of the system depicted by (1), we derive a closed form solution for X_i. We begin by rewriting (1) as follows:

$$X_{i+1} = X_i(2 - X_i) \quad \text{for } i \ge 0 \tag{2}$$

Equation (2) is the familiar logistic equation [15] with growth parameter $r = 2$. If we set $Y = X/2$, we can rewrite (2) in its canonical form as follows:

$$Y_{i+1} = 2Y_i(1 - Y_i) \quad \text{for } i \ge 0 \tag{3}$$

The dynamic properties of (3) have been well studied [16]. It has been shown that for the growth parameter $r > 3$, the equation exhibits chaotic behavior. Here we have $r = 2$ and thus the equation is stable and has the steady state value of $Y^* = 1/2$ (or $X^* = 1$). For $Y_0 < 1/2$, Y_i is monotonically increasing for $i \ge 0$, until convergence is attained. While closed form solutions exist for logistic differential equations for generic values of *r*, such solutions exist for only a handful of specific values of *r* for logistic difference equation. For $r = 2$ (3), the closed form solution is given by the following for $i \ge 0$.

$$\begin{aligned} Y_i &= \frac{1}{2}\left[1 - e^{2^i \ln(1-2Y_0)}\right] \\ \equiv\ X_i &= \left[1 - e^{2^i \ln(1-X_0)}\right] = \left[1 - (1 - X_0)^{2^i}\right] \end{aligned} \tag{4}$$

Assuming portion of the nodes that initially have the message is *1/N,* we get:

$$X_i = 1 - (1 - \frac{1}{N})^{2^i} \tag{5}$$

To approximate the number of rounds in which X converges we set $\varepsilon = \frac{1}{N}$. Since $\frac{1}{N} \le \ln 2 \cdot \log \frac{N}{N-1} \le \frac{1}{N-1}$ total number of rounds, τ, can be obtained from:

$$\begin{aligned} \tau &= \log\log N - \log\log \frac{N}{N-1} \\ &< \log N + \log\log N + \ln 2. \end{aligned} \tag{6}$$

Throughout this paper, $\log N$ refers to logarithm base 2.

Clearly, for large *N*, right hand side of this inequality is a very good estimate for the total number of rounds for pull algorithm. Comparing (6) with the required number of rounds for push&pull algorithm, $\log_3 N + O(\ln\ln N)$ [3], and push algorithm, $\log N + \ln N + O(1)$ [2], confirms the accuracy of our estimate. As expected, number of rounds required for convergence of pull algorithm is greater than push&pull but is less than push.

We can also obtain an expression for the normalized total communication cost by adding the number of calls in each round, $1 - X_i$, for τ rounds using the following[17]:

$$S_\tau = \sum_{i=0}^{\tau}(1 - X_i) = \sum_{i=0}^{\tau}(1 - \frac{1}{N})^{2^i} \approx \log N - 0.3327 \tag{7}$$

III. PUSH DIFFUSION MODEL

Let X_i be the proportion of the informed nodes in the network in round *i*. Then,

$$E(1-X_{i+1} \mid X_i) = (1-X_i)(1-\frac{1}{N})^{NX_i} \quad \text{for } i \geq 0.$$

Again, ignoring the fluctuation of X_{i+1} around its mean value, for large *N*, we have the following approximate progression for the process X_i:

$$X_{i+1} = 1-(1-X_i)e^{-X_i} \quad \text{for } i \geq 0 \tag{8}$$

As mentioned in push algorithm it is possible for more than one informed nodes to call the same uninformed node. Probability of such collisions leads to a more complex non-linear difference equation for progression of informed nodes in push algorithm, making it hard to obtain a closed form solution for (8) and the associated cost function. Therefore, we mainly rely on simulations to compare the performance of push and pull algorithms.

Fig. 1 shows how a single gossip propagates throughout the network for *N=2^15* for both push and pull algorithms. We observe that, compared to push algorithm, pull algorithm converges faster. As the number of informed nodes increases, probability of more than one informed nodes calling the same uninformed node increases which results in slowing down of convergence in push algorithm. Another major difference between push and pull algorithms is in their start and stop time. In pull algorithm it is not clear when the nodes need to start pulling information from each other. Therefore, pull algorithm is more suitable for applications where updates occur frequently. The stopping time, on the other hand, is trivial since the algorithm terminates when all the nodes are informed. In contrast, push algorithm starts when a node, or a subset of nodes, acquire a new message. However, the algorithm is very sensitive to stopping time. Stopping the information diffusion too early could leave many nodes uninformed and not stopping the algorithm at the right time results in a large increase in communication cost.

Assuming an exact estimate of the right termination time, communication cost of the push algorithm is less than pull for large *N*. A fair comparison however will have to consider the termination mechanism used in push. Fig. 2 depicts the normalized communication cost as a function of *N* for both push and pull algorithm. As expected, from (7), the normalized cost of pull exhibits a logarithmic relationship with respect to *N* and is higher than push for *N>64*.

IV. FIRST-PUSH-THEN-PULL DIFFUSION MODEL

Our *FPTP* gossip algorithm is based on the simple fact that although growth rate of the informed nodes in push and pull algorithms in the early rounds are quite comparable, the communication cost of push is much less than pull. The opposite is true in the late rounds of gossip algorithm before every node is informed. Therefore, a cost-efficient gossip algorithm needs to combine the advantages of push and pull algorithms. One possible approach is to begin with push algorithm and then at some point switch to pull algorithm.

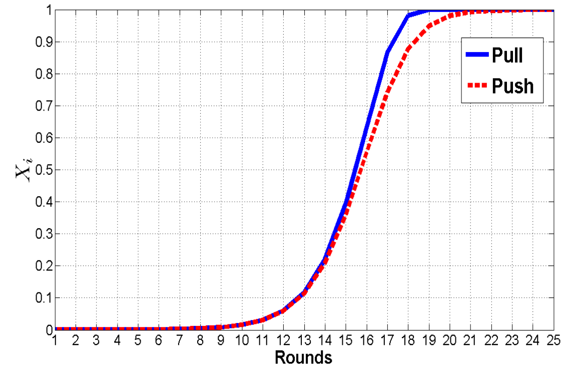

Figure 1. Gossip growth comparison between Push and Pull algorithms for *N=2^15*.

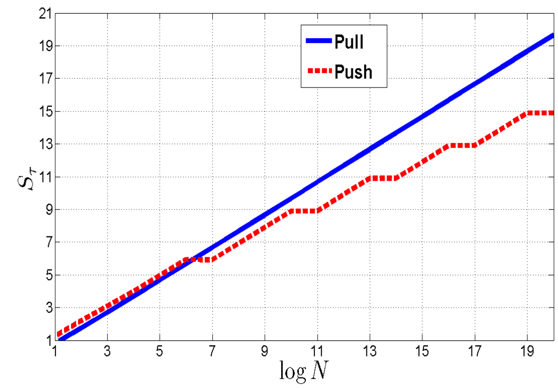

Figure 2. Communication cost comparison between Push and Pull algorithms as a function of *N*.

Clearly, the difficulty is how to determine the transition round, τ_s, for switching from push to pull. In this section we show that minimum communication cost is achieved if we switch from push to pull at round $\tau_s = \text{Round}(\log N)$. Furthermore, we show that the minimum normalized communication cost is almost constant for large *N*.

Theorem *For the First-Push-Then-Pull algorithm, normalized communication cost is* ***minimized*** *to approximately a* ***constant*** *when* $\tau_s = \text{Round}(\log N)$.

Proof. We begin by showing that the minimum cost occurs when $\tau_s^* = \text{Round}(\log N)$. First we note that the total cost is a polynomial in proportion of informed nodes at round τ_s which monotonically increases with τ_s. Applying Descartes' rule of signs to the derivative of this polynomial, it follows that *FTPT* algorithm has exactly one real minimum as a function of transition round τ_s. The rest of the proof is by induction on log *N*, where the base case is an arbitrary large *N*. For a fixed large N, N_0, it is easy to show numerically that the theorem holds. For the induction step, assume that the theorem is true for a large number of nodes N_1. Then, for $N_2 = 2N_1$, since $X_1 = 2/N_2 = 1/N_1$ with very high probability, the minimum communication cost occurs when $\tau_s = \text{Round}(\log N_1) + 1 = \text{Round}(\log N_2)$.

Once we establish that the minimum cost always occur at $\tau_s = \text{Round}(\log N)$, it becomes clear from the induction steps that X_{τ_s} is approximately constant for all large N which immediately leads to a constant minimum cost. ■

Fig. 3 shows the communication cost as a function of transition round for several values of *N*. As expected, the minimum normalized cost is constant. The constant normalized communication cost for the optimum *FPTP* algorithm is approximately 2.606 transmissions/message/node. The proportion of informed nodes at the transition round is also constant and is approximately $X_{\tau_s} = 0.3593$. Furthermore, the number of rounds for *FPTP* algorithm to converge is $\tau = \log N + 4$ which is smaller compared to push algorithm for large *N* but is less than the number of rounds for pull algorithm only when *N>2^16*.

Note that since *FPTP* starts with push algorithm and ends with pull algorithm, nodes don't need to have an estimate of starting/stopping time but need to have an estimate of the transition round. However, only uninformed nodes need to know when to start pulling. The switching mechanism is a design issue which is outside the scope of this paper.

V. LIMITTED CONNECTIVITY/COOPERATION CASE

We now extend our basic model to allow nodes to limit their cooperation or allow for limited connectivity between the nodes. Here, we examine the effect of probability of cooperation/connectivity between the nodes of a network on the overall cost and the rate of spread of new messages. We introduce a new parameter α, to represent a constant/average probability of connection/cooperation. Then, the information diffusion model for pull algorithm can be described by the following recursion:

$$X_{i+1} = \alpha X_i(1 - X_i) + X_i \quad \text{where} \quad \alpha \in (0,1]. \tag{9}$$

Similarly, the information diffusion model for push algorithm is given by the following recursion:

$$X_{i+1} = 1 - (1 - X_i)e^{-\alpha X_i} \quad \text{where} \quad \alpha \in (0,1] \tag{10}$$

As mentioned a closed form solution for (9) is non-trivial making it difficult to analytically examine *FPTP* performance. However, following the same analysis in Section IV, it is still possible to show that *FPTP* has a single minimum cost as a function of transition round for a given α and that it is approximately constant. Fig. 4 shows the normalized communication cost as a function of transition round for several values of α for *N=2^15*. A most interesting observation in Fig. 4 is that the optimum transition round occurs when the transition round is equal to the total communication cost of pull algorithm (i.e. $\tau_s = S_0$). Furthermore, as we might expect the minimum cost increases as the connection probability decreases.

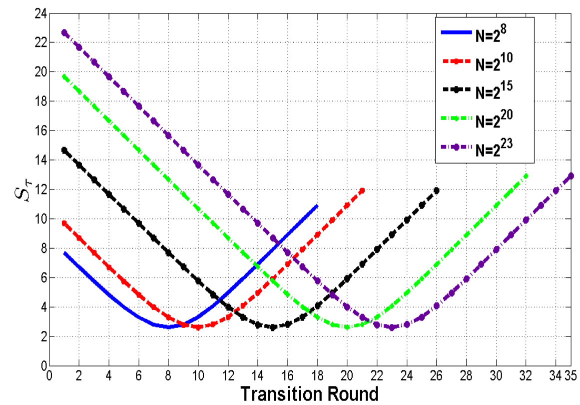

Figure 3. Normalized communication cost for *FPTP* algorithm as a function of transition round.

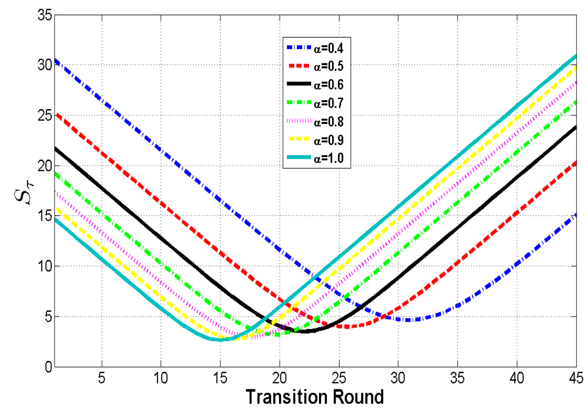

Figure 4. Normalized communication cost for *FPTP* as a function of transition round for several values of α (*N=2^15*).

Fig. 5 shows the normalized communication cost of *FPTP* algorithm for several values of *N* with $\alpha = 0.6$. We observe that, minimum cost is approximately constant at about 3.5 and the optimum transition round is $\log N + 7$ which is again the same as the total communication cost for pull algorithm (i.e. $\tau_S = \text{Round}(S_0)$).

VI. MOBILE AD-HOC NETWORK APPLICATION

In this section we apply our *FPTP* algorithm to MANET networks to examine our results in the context of information dissemination in mobile ad-hoc networks. We use random-waypoint model [18] in our simulations to model mobility. In random-waypoint model a node chooses a point uniformly at random within the grid and moves toward it with a random speed. Once the node gets to its destination it pauses for some time and then repeats the process. To simulate limited connectivity, we assume a communication range for all the nodes beyond which the message relay fails. The networks under consideration are fully distributed. That is, nodes have no knowledge of the other nodes' state (e.g. location, informed/uninformed state, etc.). Therefore, any communication attempt, successful or unsuccessful, adds to the communication cost.

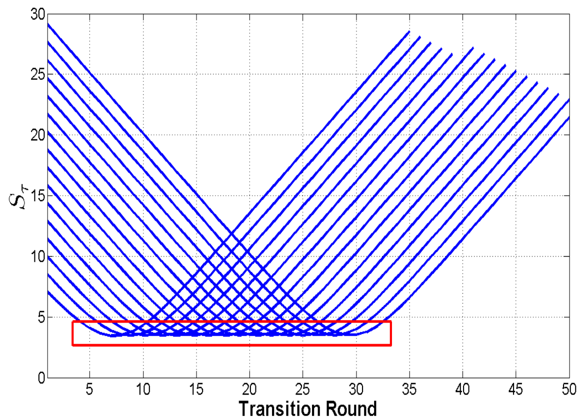

Figure 5. Normalized communication cost for *FPTP* algorithm as a function of transition round for $\alpha = 0.6$.

The parameters for the simulation environment are set as follows:

- Area: 100X100 meters
- Average Speed: 3 meters/second
- Pause time: 1 second
- Transition round from push to pull: $\log N + 7$
- 1000 runs for each case
- Rounds interval: 1 second

The transition round was chosen to minimize the communication cost. Initially, all the nodes are placed uniformly at randomly within the area. In our first experiment the communication range is set to the average distance of all the nodes which corresponds to approximately 50% chance of connectivity.

Table-1 shows the communication cost of disseminating a single message in transmissions/node for $N = 128, \cdots, 2048$. As expected, the overhead for push and pull algorithms increase logarithmically with the number of nodes. In contrast, for the *FPTP* algorithm increasing the number of nodes results in a small fluctuation in the communication overhead. The second observation is that increase in the number of rounds required for all the nodes to become informed is regular (approximately 2 rounds when $\log N$ increases by one. It is also interesting to note that the communication cost is less than our analytical results. One contributing factor could be that in the random-waypoint mobility model the distribution of the nodes changes over time and the nodes gravitate toward the center [19]. Therefore, node distribution does not remain uniform and the connection probability increases over time resulting in faster convergence and lower cost.

In the second experiment we simulated propagation of two messages throughout the network. We again assumed limited communication range for all the nodes. Initially, two randomly selected nodes are assigned messages 1 and 2 respectively. Table-2 shows the communication cost of disseminating two

TABLE I. COMMUNICATION COST IN TRANSMISSIONS/NODE (SINGLE MESSAGE CASE)

		NUMBER OF NODES				
		128	**256**	**512**	**1024**	**2048**
FPTP	***Cost (ave)***	**2.793**	**2.658**	**2.532**	**2.553**	**2.478**
	Rounds (ave)	**21.168**	**23.076**	**25.974**	**28.462**	**31.181**
PULL	***Cost (ave)***	**9.706**	**10.687**	**11.386**	**12.478**	**13.327**
	Rounds (ave)	**19.204**	**21.456**	**23.408**	**25.820**	**28.240**
PUSH	***Cost (ave)***	**11.845**	**13.353**	**14.737**	**16.494**	**18.164**
	Rounds (ave)	**23.014**	**25.796**	**28.372**	**31.492**	**34.311**

TABLE II. COMMUNICATION COST IN TRANSMISSIONS/NODE (TWO MESSAGE CASE)

		NUMBER OF NODES				
		128	**256**	**512**	**1024**	**2048**
FPTP	**Cost (ave)**	**6.690**	**6.318**	**6.147**	**6.081**	**5.783**
	Rounds (ave)	**32.124**	**34.900**	**36.201**	**38.220**	**41.522**
PULL	**Cost (ave)**	**14.583**	**15.881**	**16.980**	**18.236**	**18.282**
	Rounds (ave)	**32.410**	**37.150**	**40.465**	**44.400**	**47.347**
PUSH	**Cost (ave)**	**24.762**	**28.334**	**31.177**	**34.744**	**37.808**
	Rounds (ave)	**34.014**	**39.280**	**43.220**	**48.260**	**51.508**

messages to the entire network in number of transmissions/node for $N = 128, \cdots, 2048$. All the simulation parameters were the same as single message case including the transition round.

The additional complication in the multiple message case stems from the fact that the nodes have no knowledge about the other nodes' message profile. Therefore, nodes are not able to efficiently decide which message to relay if they have both messages. Furthermore, each node can only send one message per round. Based on these assumptions, if a node possesses more than one message, it has to randomly pick one to relay [20]. This is a source of additional inefficiency that leads to increased cost and latency. Our simulations show that for the *FPTP* gossip algorithm increase in the number of transmissions per message is minimal.

The overhead for the push and pull gossip algorithms again increase logarithmically with the number of nodes. For the *FPTP* gossip algorithm, the overhead stays relatively constant and, as expected, the communication cost is slightly larger than twice the cost of single message case. Although the communication cost almost doubles, number of rounds doesn't necessarily follow the same trend since there are more communications taking place in each round.

There are few interesting observations in these results some of which are expected and some are counter intuitive and need further investigation. First is the optimum transition round from push to pull algorithm which remains the same as the single message case. Although the optimum transition round is still $\log N + 7$, the communication cost is not as sensitive to the transition round as in the single message case. Second, the communication cost for push algorithm is much higher than the pull algorithm. This seems to be consistent with the lower transition round to pull algorithm which is a result of faster growth of informed nodes. Third, the *FPTP* algorithm becomes more efficient as the number of nodes increase. A limited number of simulations have shown that as the number of nodes increase the communication cost approaches twice the cost of single message case.

VII. CONCLUDING REMARKS

We introduced a novel gossip algorithm for cost-efficient dissemination of new information in decentralized networks. We demonstrated that in a fully connected and cooperative network optimum combination of push and pull algorithms can drastically reduce the communication cost. Our *FPTP* gossip algorithm minimizes the expected normalized communication cost to a constant which we believe can be used as a lower bound on the number of required transmissions in gossip algorithms. We also showed that the same results hold for the networks with limited connectivity. We then validated our results by testing our algorithm in a MANET network environment using random-waypoint mobility model and demonstrated that the simulation results closely follow our analysis.

There is considerable scope for further research that we are currently exploring. One open question is whether the constant overhead derived in this paper is the lower bound on the communication cost of all variants of gossip algorithm. Another interesting problem is the relationship between the optimum transition round in the limited connectivity case and the connection probability α . Our observation that the optimum transition round is approximately equal to the total cost for the pull algorithm is most intriguing.

Another aspect concerns analysis of the multiple message case. Our simulations raised number of interesting questions that we need to gain a better understanding about them. Most notable among them is the optimum transition stage which remained the same as the single message case although the number of rounds increased substantially. Another extension is to formulate our algorithm for arbitrary number of messages and examine the results in relations to increasing number of messages.

Finally, we plan to investigate a repertoire of MANET routing algorithm that include various communication modes such as broadcast and directional transmissions. It is also interesting to investigate the performance of *FPTP* gossip algorithm for different mobility models and compare the results to the more common flooding approaches.

REFERENCES

[1] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry. Epidemic algorithms for replicated database maintenance. In PODC 1987, pages 1-12.B.

[2] Pittel, “On spreading a rumor”, SIAM Journal on Applied Mathematics, 47(1):213–223, 1987.

[3] R. Karp, C. Schindelhauer, S. Shenker, and B. V¨ocking. Randomized rumor spreading. Proceedings of 41st Symposium on Foundations of Computer Science (FOCS), 2000, pp. 565–574.

[4] D. Agrawal, A. El Abbadi, and R. C. Steinke, “Epidemic algorithms in replicated databases (extended abstract),” in Proceedings of the 16th Symposium on Principles of Database Systems (PODS), 1997, pp. 161–172.

[5] David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In 44th Annual IEEE Symposium on Foundations of Computer Science, Cambridge, MA, USA, 2003. IEEE Computer Society.

[6] R. van Renesse, Y. Minsky, and M. Hayden. A gossip-style failure detection service. In Proc. of IFIP Int-l Conference on Distributed Systems Platforms and Open Distributed Processing, pages 55-70, 1998.

[7] Van Renesse, R., Birman, K.P., Vogels, W.: Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. ACM Transactions on Computer Systems 21(2) (2003)164–206.

[8] C. Gkantsidis and P. Rodriguez, “Network coding for large scale content distribution,” in Proceedings of the 24th International Conference on computer Communications (INFOCOM), vol. 4, 2005.

[9] L. Ding and P. Manoj. On randomized broadcasting and gossiping in radio networks. In Proceedings of the 8th Annual International Conference on Computing and Combinatorics, pages 340–349, 2002.

[10] P. Eugster, R. Guerraoui, S. Handurukande, A-M Kermarrec, and P. Kouznetsov. Lightweight probabilistic broadcast. ACM Trans. on Computer Systems, 21(4), 2003.

[11] Z. J. Haas, J. Y. Halpern, and L. Li. Gossip-based ad hoc routing. IEEE/ACM Transactions on Networks (TON), vol. 14, no. 3, pp. 479–491, 2006.

[12] Frey, D., Guerraoui, R., Kermarrec, A.-M., Monod, M., Qu´ema, V.: Stretching Gossip with Live Streaming. In: Proc. of DSN, 2009.

[13] R. Bakhshi, L. Cloth, W. Fokkink and B. Haverkort. Mean-field analysis for the evaluation of gossip protocols. In Proc. of 6th International Conference on Quantitative Evaluation of SysTems (QEST), Pages 247-256, IEEE Computer Society, 2009.

[14] P. T. Eugster, R. Guerraoui, A.-M. Kermarrec, and L. Massouli´e. Epidemic information dissemination in distributed systems. IEEE Computer, 37(5):60–67, 2004.

[15] Robert M. May, “Simple mathematical models with very complicated dynamics,” Nature, 261:459–467, 1976.

[16] Ali Saidi, Mahesh Tripunitara, Mojdeh Mohtashemi. Social Learning Applications in Resource Constrained Networks. IEEE International Conference on Social Computing, August 29-31 2009.

[17] Ali Saidi and Mojdeh Mohtashemi. A new algorithm for computing logarithm. In prepration. 2011.

[18] J. Yoon, M. Liu, and B. Noble, “Random waypoint considered harmful,” in *Proc. IEEE INFOCOM*, 2003, pp. 1312–1321.

[19] C. Bettstetter, G. Resta, and P. Santi, “The node distribution of the random waypoint mobility model for wireless ad hoc networks,” *IEEE Trans. Mobile Comput.*, vol. 2, no. 3, pp. 257–269, Jul.-Sep. 2003.

[20] S. Deb, M. M´edard, and C. Choute, “Algebraic gossip: A network coding approach to optimal multiple rumor mongering,”*IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2486–2507, June 2006.