Table 1: Information retrieval and dissemination with a creation rate of 25 blogs per second for various network sizes - μ_a is the average retrieval accuracy, μ_r is the average replication rate (% of network), and σ denotes the corresponding standard deviation.

$n = 10,000$					$n = 100,000$					$n = 500,000$				
$\lvert\mathcal{T}\rvert$	Retrieval		Dissemination		$\lvert\mathcal{T}\rvert$	Retrieval		Dissemination		$\lvert\mathcal{T}\rvert$	Retrieval		Dissemination	
	μ_a	σ_a	μ_r	σ_r		μ_a	σ_a	μ_r	σ_r		μ_a	σ_a	μ_r	σ_r
1	90.652	0.421	7.371	3.752	2	93.509	0.180	11.807	6.518	2	92.044	0.312	11.647	6.652
2	**97.045**	**0.177**	**13.055**	**6.236**	**3**	**96.676**	**0.147**	**16.635**	**8.699**	**3**	**95.657**	**0.154**	**16.446**	**8.903**
3	98.701	0.114	18.058	8.182	4	98.074	0.130	20.947	10.468	4	97.383	0.223	20.700	10.606

the entire network. Both assumptions are arbitrary. The decision to query 25 nodes reflects the need to minimize network bandwidth and latency for the user. Setting $\alpha = 0.95$ and $z = 25$, we get $\frac{r}{n} = 0.12$, i.e. a blog needs to be replicated to 12% of the network. The accuracy measure applies to both the retrieval of the *followed* blogs as well as to searching the contents of all the blogs in the network.

Next, we assume that each node in the network contributes, on average, 1GB of disk space to support the services. Of this contributed disk space, 90% is utilized for storage of the blogs, and the remaining 10% for indexing the stored blogs.

When a node queries other nodes, or is queried by another node, it transfers its most recent blog, thereby spreading the blog into the network. To increase the rate of spreading, we introduce a *transfer buffer*, $\mathcal{T}$, which stores a small fraction of the most recent blogs that the peer has encountered. The transfer buffer has a fixed size, $\mathcal{T}_L$. To retrieve the blogs Node $\mathcal{A}$ is *following*, it makes a *blog-request* to z other nodes every s seconds. A request consists of 1) the list, $\mathcal{L}_A$, of the user ID's Node $\mathcal{A}$ follows 2) the most recent blog created at Node $\mathcal{A}$, and 3) the blogs contained in the *transfer buffer* $\mathcal{T}_A$ of Node $\mathcal{A}$. When Node $\mathcal{B}$ receives a *blog-request*, it sends back a *blog-response*, which consists of 1) the most recent blogs of all the peers specified in $\mathcal{L}_A$, which are found in its storage area, 2) the most recent blog created at Node $\mathcal{B}$, and 3) the blogs contained in its *transfer buffer* $\mathcal{T}_B$.

To perform a keyword search, the list $\mathcal{L}$ is replaced by the keyword query in the *request*, and the *response* contains the blogs matching the query, which are then merged and re-ranked at the originating node.

At the completion of a request/response, the node stores all the new blogs it has come across into it's storage area. It then reconstructs its transfer buffer. A number of strategies for selecting the blogs for the buffer are possible. We investigated two, namely most recent, and a probabilistic strategy in which the probability of selection is proportional to the age of the blog. The performance of the selection methods was nearly identical with the deterministic method performing marginally better. We omit the results for the probabilistic methods due to space limitations. The goal of selecting blogs for the transfer buffer is to continue spreading newer blogs at the expense of older blogs.

3. SIMULATIONS

To verify our theoretical framework, we performed simulations based upon the parameter values discussed in the previous section. At each iteration, 25 new blogs are created at random nodes in the network. This is the normalized rate of blog creation for a network of 1M nodes based upon Twitter's average of 2,300 tweets per second and a 100 million *active* user base The simulations are based on network sizes of 10,000, 100,000, and 500,000 nodes. Since z is small, we assume that each iteration is one second in magnitude, i.e. the latency is less than 1 second, and that iteration and second can be used interchangeably. Under steady-state conditions, i.e. when all transfer buffers are full, we recorded each blog's replication rate at the end of each simulation. Retrieval accuracy was also measured. We assumed each node follows 10 randomly chosen user ID's.

Table 1 summarizes the results. For blogging rates of 25 per second, we observe that a buffer size of 2 or 3 is sufficient to provide accuracies of 95% or higher. The average replication rate was slightly higher than the required 12%, but did not exceed 17%. The mean and standard deviations are the average over the 10 trials, for each configuration.

4. CONCLUSIONS AND FUTURE WORK

This paper considered the design of a micro-blogging social network in an unstructured peer-to-peer network. It consists of a PAC search together with a constrained rumour spreading algorithm. Simulations showed that the rumour spreading algorithm successfully spreads blogs to a fraction of the network (about 12%), which is sufficient to guarantee that the retrieval accuracy is at least 95%. This was achieved with a small transfer buffer size of just 2 or 3.

Our current on-going work focuses on churn. Since unstructured networks do not have to maintain a DHT, the effect of churn (i.e. nodes joining and leaving) on the framework would be to increase the number of nodes queried or the replication rate to achieve the same retrieval accuracy.

Analysis of Twitter shows that the blog creation rate of individual users is not constant, but follows a power law distribution. Similarly, the *follower* and *followed by* statistics also follow a power law distribution. Furthermore, the blog creation rate contains spikes where the rate increases five fold when an important event occurs. We plan to incorporate these statistics into our framework and use them to arrive at a system-wide accuracy metric as part of our future work. From our experiments, it is clear that the required size of the transfer buffer depends on the network size and the blog creation rate. We intend to investigate an *adaptive* transfer buffer strategy where a node would select the size of the transfer buffer based upon its local knowledge of the network size, blog creation rate, and an estimate of the replication rate of blogs.

5. REFERENCES

[1] Asthana, H., Fu, R., and Cox, I. J. On the feasibility of unstructured peer-to-peer information retrieval. In *ICTIR* (2011).

[2] Bortnikov, E., Gurevich, M., Keidar, I., Kliot, G., and Shraer, A. Brahms: Byzantine resilient random membership sampling. *Computer Networks 53*, 13 (2009), 2340–2359.

[3] Cox, I. J., Fu, R., and Hansen, L. K. Probably approximately correct search. In *ICTIR* (2009).

[4] Demers, A., Greene, D., Hauser, C., Irish, W., Larson, J., Shenker, S., and Sturgis, H. Epidemic algorithms for replicated database maintenance. In *Proceedings of the 6th annual ACM Symposium on Principles of distributed computing* (1987).

[5] Urdaneta, G., Pierre, G., and Steen, M. A survey of dht security techniques. *ACM Computing Surveys (CSUR) 43*, 2 (2011), 8.