Bloom Filters and Overlays for Routing in Pocket Switched Networks

Christoph P. Mayer
Institute of Telematics, University of Karlsruhe, Germany
mayer@tm.uka.de

ABSTRACT

Pocket Switched Networks (PSN) [3] have become a promising approach for providing communication between scarcely connected human-carried devices. Such devices, e. g. mobile phones or sensor nodes, are exposed to human mobility and can therewith leverage inter-human contacts for store-and-forward routing. Efficiently routing in such delay tolerant networks is complex due to incomplete knowledge about the network, and high dynamics of the network. In this work we want to develop an extension of Bloom filters for resource-efficient routing in pocket switched networks. Furthermore, we argue that PSNs may become densely populated in special situations. We want to exploit such situations to perform collaborative calculations of forwarding-decisions. In this paper we present a simple scheme for distributed decision calculation using overlays and a DHT-based distributed variant of Bloom filters.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design —*Distributed networks, Store and forward networks*

General Terms

Routing, Data structures

Keywords

Pocket Switched Network, Bloom Filter, Overlay

1. BLOOM FILTERS FOR PSN ROUTING

Standard Bloom Filters.

We propose the use of Bloom filters [1] for space-efficient routing in PSNs, as they can encode a device's contact graph and therewith be used to calculate good forwarding strategies. A Bloom filter is a probabilistic and space-efficient data structure for efficient lookups. It consists of a bit array B of size m. Using k different hash functions $h_i(x) \mapsto [0, m-1]$ elements $x \in S$ are hashed to k different positions $d_i = h_i(x)$ and the array B updated in the corresponding positions: $B[d_i] = 1$. Correspondingly, the lookup of an element $x \in S$ checks whether all k bits $B[d_i]$ are set to 1. A Bloom filter has no false negatives ($x \in B \rightarrow \forall k : B[d_i] = 1$), but can have false positives ($x \notin B \nrightarrow \forall k : B[d_i] = 0$).

CoNEXT'09 Student Workshop, December 1, 2009, Rome, Italy.
ACM 978-1-60558-751-6/09/12.

Basic Forwarding on Bloom Filters.

We assign every device D_j an identifier ID_j (e. g., MAC address, random/cryptographic identifier). Furthermore, one Bloom filter B_j that represents contact information with other devices is stored per device D_j. When two devices D_j and D_k meet, first, each device updates its Bloom filter with the other device identity ($insert(B_j, ID_k)$, and $insert(B_k, ID_j)$). Second, each device requests the Bloom Filter of the other device. For each message stored on D_j for forwarding, D_j checks whether D_k has contact information in its Bloom filter for the destination messages stored on D_j, and forwards messages accordingly. Using this scheme contact information is collected and used for message forwarding in a space-efficient but non-probabilistic way.

Fuzzy Bloom Filters.

Standard Bloom filters answer membership queries in a binary *yes/no* style. We now extend Bloom filters to answer membership queries in probabilistic *0%–100%* ways to design a resouce-efficient PRoPHET [5]-like scheme. Therefore we define device-specific probabilities $p \in P, p_i \in [0, \dots, 1]$ so that for each $x_i \in S$ the contact probability is $P(x_i) = p_i$. When inserting x_i into the Bloom filter using p_i, we no longer set k bits in B to 1 but rather $k \cdot p_i$ bits with $0 \leq (k \cdot p_i) \leq k$. If $\cup$ and $\cap$ operations are required on the Fuzzy Bloom filters, we can't set random $k \cdot p_i$ bits but are required to start filling up the $k \cdot p_i$ bits e. g. from low to high. Membership queries can now be answered by the Bloom filter with probabilities that are calculated using the fraction of actual bits set. The granularity of p_i is directly constrained on the number of hash functions k. Deriving the false-positive rate and calculating the optimal size of B is subject to current research.

Aging Fuzzy Bloom Filters.

We can now perform 'aging' of all $x_i \in S$ that have been inserted into the Fuzzy Bloom Filter by resetting bits in B. Therefore, in each timestep t (e. g. one day) we set a small percentage of random bits of B_j from 1 to 0. This way the membership probabilities returned by the Fuzzy Bloom Filter now decrease over time. Using this scheme, a highly resource-efficient probabilistic routing in PSNs can be achieved that is applicable even to small sensor nodes.

Figure 1 shows an example of such a fuzzy Bloom filter

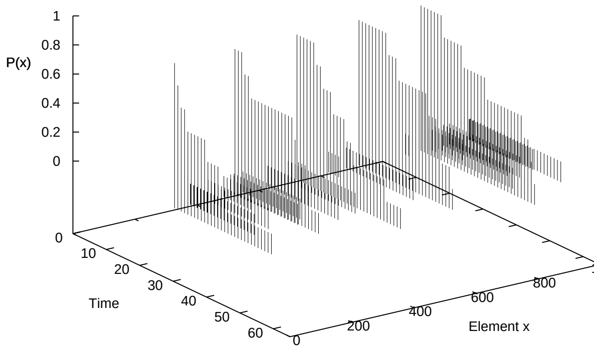

Figure 1: Fuzzy Bloom filter with aging entries

with aging entries using 7 hash functions. By using 7 hash functions the granularity of $p \in P$ is restricted to multiples of $\sim 14\%$. At timestep 30 we insert $x_0, \cdots x_4$ into the Bloom filter. Each inserted x_i has initially 100% of required bits set to 1 and therewith a probability of 1.0 returned by the *lookup* function. In each timestep 1% of random bits set to 1 are reset, resulting in the $P(x_i) = p_i$ to decrease. Using this mechanism the p_i in the Bloom filter become fuzzy over time and vanish. Using this meachanism probabilities in the Bloom filter of contact encounters are dynamic a can be used to implement resource-efficient PSN routing.

2. COLLABORATIVE FORWARDING

PSNs are considered scarcely connected but dense PSNs can occur, e. g. in mass transportation like trains, or city centers. Detecting and exploiting such situations can possibly improve routing efficiency. We propose to build up spontaneous overlay networks of devices—e. g. traveling together in a train—to collectively calculate forwarding decisions in a distributed and scalable fashion. We build upon our extended Bloom filter schemes detailed in Section 1.

In case of large numbers of devices a naive scheme that exchanges all Bloom filters between all devices is not scalable. To scalably perform collaboration we use a scheme similar to Distributed Bloom filters [4]: All Bloom filters are split into segments of equal size. Segments are distributed using a segment index in a DHT that all devices take part in. Each device that stores a specific segment of all Bloom filters performs logical OR operations to merge segments.

A membership query can now use the DHT to resolve whether any of the devices taking part in the overlay has contact information for a given x_i. Therefore we apply the k hash functions to calculate k indexes into the Bloom filter. For each of the indexes we calculate the segment index and send queries into the DHT using segment index, and bit index inside this segment. This way, k requests are send into the DHT in parallel to perform queries of specific bits. Using the Fuzzy Bloom filters as described in Section 1 probabilistic membership queries can be performed in a distributed and scalable way, where *only the highest peering probability is returned by the DHT*. Furthermore, if a specific probability threshold for p_i is required, testing the highest k'th bit first will reduce message overhead. This way a device can quickly find out whether a participating device knows x_i with a probability of e. g. $\geq 80\%$. In case the required highest bit is not set, a quick rejection is possible.

If a match has been found in the DHT the exact node needs to be determined that contributed the specific parts of the Bloom filter. This information can only be determined by the device-specific Bloom filters. In a first approach, a broadcast scheme is used to find the device that contributed the specific bits. Such broadcast requests can be answered by receiving devices very quickly, as they can start checking bits from top to bottom, starting with h_{k-1}. This way they can easily ignore the request if the first bit from the top hash functions is not set. A device that successfully checked all required bits of the hash functions $h_{k-1} \dots h_0$ will finally answer the querying device.

Using the presented scheme cooperation between devices can be implemented scalably and possible forwarding devices with the highest peering probability can be detected quickly. For actual message forwarding the specific device needs to be detected that contributed this highest peering probability. A first approach is to use a broadcast scheme with quick rejection at the receiving devices.

3. CONCLUSION

In our research we are looking into resource-efficient routing for Pocket Switched Networks using Bloom filters and the collaborative calculation of message forwarding strategies using spontaneous overlay networks. The goal is to encode as much of the contact graph as possible to provide quality forwarding decisions and efficient message routing, with focus on resource-constrained devices. We are currently working on evaluations and aim to integrate the presented scheme into the Ariba platform [2]. Furthermore, handling churn in the overlay is critical—as devices are exposed to human mobility—and subject to future research.

Acknowledgments

The author thanks Oliver P. Waldhorst for helpful comments and suggestions. This work was funded as part of the *Spontaneous Virtual Networks* (*SpoVNet*) project by the Landesstiftung Baden-Württemberg within the BW-FIT program.

4. REFERENCES

[1] B. H. Bloom. Space/time Trade-offs in Hash Coding with Allowable Errors. *Communications of the ACM*, 13(7):422–426, July 1970.

[2] C. Hübsch, C. P. Mayer, S. Mies, R. Bless, O. P. Waldhorst, and M. Zitterbart. Reconnecting the Internet with ariba: Self-Organizing Provisioning of End-to-End Connectivity in Heterogeneous Networks. In *Proceedings of ACM SIGCOMM*, Barcelona, Spain, Aug. 2009. Demo.

[3] P. Hui, A. Chaintreau, R. Gass, J. Scott, J. Crowcroft, and C. Diot. Pocket Switched Networking: Challenges, Feasibility, and Implementation Issues. In *Proceedings of the Workshop on Autonomic Communications*, pages 1–12, Athens, Greece, Oct. 2005.

[4] C. Jamard, G. Gardarin, and L. Yeh. Indexing Textual XML in P2P Networks Using Distributed Bloom Filters. In *Proceedings of DASFAA*, pages 1007–1012, Bangkok, Thailand, Apr. 2007.

[5] A. Lindgren, A. Doria, and O. Schelén. Probabilistic Routing in Intermittently Connected Networks. *SIGMOBILE Mobile Computing Communications Review*, 7(3):19–20, July 2003.