HAL
archives-ouvertes.fr

A hash-based paging and location update procedure

Pars Mutaf, Claude Castelluccia

▶ **To cite this version:**

Pars Mutaf, Claude Castelluccia. A hash-based paging and location update procedure. WiOpt'03: Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, Mar 2003, Sophia Antipolis, France. 4 p., 2003. <inria-00466582>

HAL Id: inria-00466582
https://hal.inria.fr/inria-00466582

Submitted on 24 Mar 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Hash-Based Paging and Location Update Procedure

Pars Mutaf
Planète Team
INRIA Rhône-Alpes
pars.mutaf@inria.fr

Claude Castelluccia
Planète Team
INRIA Rhône-Alpes
claude.castelluccia@inria.fr

Abstract

We propose a hash-based paging and location update procedure that reduces the paging cost in cellular systems. By applying a Bloom filter, the terminal identifier field of a paging message is coded to page a number of terminals concurrently. A small number of terminals may wake up and send what we call "false location updates" although they are not being paged. We compare the total number of paging and false location update messages with the cost of the normal paging procedure. The larger the size of the terminal identifier, the less probable are false location updates. Therefore, hash-based paging especially shows promise for cellular systems based on Mobile IPv6 with 128-bit mobile host addresses.

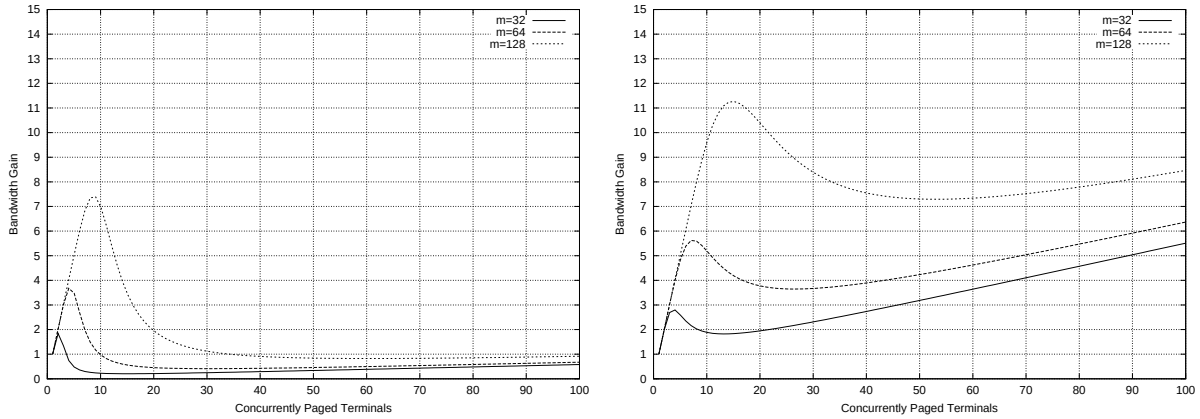
1 Introduction

In cellular systems, terminals must update the network with their current location in order to receive packets. Idle mobile terminals update the network only when they cross paging area boundaries, which cover a relatively large number of cells. Since terminals are idle most of the time, paging reduces the bandwidth cost of location updates and battery drain on terminals. The cost to pay is the bandwidth cost of broadcasting a paging message in each cell of a paging area upon call arrival. A paged terminal reports its exact location by sending a location update message and the communication can begin.

A better paging service demands larger paging areas. However, large paging areas bring a permanently high paging load (i.e. incoming call arrival rate) since a large amount of terminals are served per paging area. The paging load may also temporarily increase due to flash crowds. In order to cope with high paging load, an important portion of the available bandwidth in each cell must be allocated to paging messages. Otherwise paging processes are queued in the network and important call setup delays are incurred, which is also undesirable. This problem has received considerable attention and important research has been done for reducing the bandwidth cost of paging messages.

In this paper, we propose a hash-based paging and location update procedure, using *Bloom Filters* [1]. A Bloom filter is a randomized data structure for concisely representing a set, in order to support membership queries. The space efficiency is achieved at the cost of a small probability of false positive. We use Bloom filters to page several terminals in a same paging area concurrently, which reduces the paging bandwidth consumption and the call setup delays. We show that important bandwidth can be saved using Bloom filters and our proposal is best suitable for Mobile IPv6 [3] based cellular systems with 128-bit mobile host addresses.

The remaining sections of this paper are organized as follows: Section 2 describes the hash-coding of paging messages using Bloom filters, Section 3 presents an analytical evaluation of the bandwidth gain that can be obtained with this approach and Section 4 presents some conclusions and future work.



(a) Macro cells.

(b) Micro cells.

Figure 1: Bandwidth Gain.

2 Hash-Based Paging and Location Update

We propose using Bloom filters [1] for concurrently paging a set $A = \{a_1, a_2, ..., a_n\}$ of n terminals that reside in the same paging area. The m-bit terminal identifier I found in a paging message is coded to represent all members of the set A. The procedure requires k independent uniform hash functions, $h_1(), h_2(), ..., h_k()$ (where, $1 \leq h() \leq m$). First, all bits of I are set to 0, then for each element $a \in A$, the bits at positions $h_1(a), h_2(a), ..., h_k(a)$ in I are set to 1 (a particular bit may be set multiple times). The resulting vector I can be used to page all members of A concurrently by broadcasting a single paging message.

Upon receipt of the paging message, a given terminal b can detect if it is being paged by checking the bit positions $h_1(b), h_2(b), ..., h_k(b)$ in I. If any of them is 0, then b is certainly not being paged. Otherwise, b is being paged and should respond with a location update message. There is a small probability that the bits at positions $h_1(b), h_2(b), ..., h_k(b)$ are set although $b \notin A$. In this case, the location update response of b is useless and we call it a *false location update*. The false location update probability is $F_{lu} = (1 - e^{-kn/m})^k$ and minimized for $k = (\ln 2) \times \frac{m}{n}$, in which case it becomes $F_{lu} = (0.6185)^{m/n}$ (adapted to paging from [2]). In the following discussion we assume that k is always optimal, i.e. chosen to minimize F_{lu} regarding the number of concurrently paged terminals [1].

3 Bandwidth Gain

Using the normal paging procedure, the paging cost in each cell of a paging area is one paging message per incoming call. In hash-based paging, the paging cost in each cell is one paging message and several false location update messages per n incoming calls. Let D the terminal density, i.e. the number of terminals per cell, then the paging cost in each cell is $1 + D \times F_{lu}$ per n incoming calls. The bandwidth gain (simplified for equal sized paging and location update messages) is:

$$G = \frac{n}{1 + D \times F_{lu}} \quad (1)$$

Figure 1 shows the bandwidth gain for different identifier sizes and terminal densities. It is assumed

[1] In practice, this can be easily achieved by fixing the maximum number of hash functions k_{max} and transmitting k in paging messages. The number of hash computations will have no impact on terminal energy consumption. A terminal b can store the results of $h_1(b), h_2(b), ..., h_{k_{max}}(b)$ and use the first k results for future paging events. On the network side, the computational load due to hash computations will be self-tuning since the optimal number of hash functions decreases with increasing paging load (i.e. n).

that in macro and micro cellular environments, the terminal densities are $D = 200$ and $D = 20$, respectively [6]. It is important to note that hash-based paging will be more attractive as cell sizes get smaller and terminal identifier sizes become larger, both of which are justified by the current trend in support of increasingly important number of cellular users. Mobile IPv6 [3] defines a 128-bit *care-of-address* that identifies a mobile host in its current subnet and a global identifier called *home address* which is also 128-bit long. Paging in Mobile IPv6 networks is not yet well defined. However, hash-based paging shows promise especially for Mobile IPv6 networks with large identifier sizes.

In practice, n should satisfy the condition that $F_{lu} \simeq 0$, so that false location updates have a negligible impact on energy consumption. This roughly corresponds to the interval where $\frac{dG}{dn} \simeq 1$ in Figure 1. Tables 1-3 show the bandwidth gain that can be obtained under this constraint and for integer k values. Important bandwidth gain can be saved with negligible energy consumption overhead, especially in Mobile IPv6 based networks. For example, Table 3 indicates that if the number of concurrently paged terminals is set to $n = 9$, a bandwidth gain of $G = 8.81$ or $G = 7.40$ can be achieved at the cost of $F_{lu} = 0.001$ (which is negligible since a given idle terminal will be unnecessarily awakened only once per 1000 paging events).

			G	
n	k	F_{lu}	$D = 20$	$D = 200$
2	11	0.000459	1.981821	1.831959
3	7	0.005947	2.681099	

Table 1: $m = 32$

			G	
n	k	F_{lu}	$D = 20$	$D = 200$
2	22	0.000000	1.999992	1.999916
3	14	0.000035	2.997879	2.978927
4	11	0.000459	3.963642	3.663917
5	8	0.002134	4.795343	
6	7	0.005947	5.362197	

Table 2: $m = 64$

			G	
n	k	F_{lu}	$D = 20$	$D = 200$
2	44	0.000000	2.000000	2.000000
3	29	0.000000	3.000000	2.999999
4	22	0.000000	3.999983	3.999832
5	17	0.000005	4.999545	4.995451
6	14	0.000035	5.995759	5.957855
7	12	0.000153	6.978653	6.792231
8	11	0.000459	7.927285	7.327835
9	9	0.001078	8.810141	7.404357
10	8	0.002134	9.590686	
11	8	0.003732	10.235926	
12	7	0.005947	10.724394	

Table 3: $m = 128$

4 Conclusion

Bloom filters can be effectively used for reducing the bandwidth cost of paging. Multiple terminal identities can be coded into a single terminal identifier found in a paging message and paged concurrently. The proposed optimization reduces the rate of broadcast paging messages and queuing delays in the network, at the cost of introducing some unnecessary i.e. "false" location update messages. Despite false location updates, important bandwidth gains can be obtained with negligible impact on energy consumption. The proposed optimization is more attractive for mobility protocols that define larger identifier sizes. Mobile IPv6 defines 128-bit mobile host addresses which can be effectively exploited using hash-based paging.

The idea presented in this paper can possibly be adapted to IPv6 Neighbor Discovery [4] for concurrent address resolution, to optimize IP-layer paging in next generation cellular systems and possibly Mobile Ad Hoc Networks [5].

5 Acklowledgements

The authors would like to thank Imad Aad and the anonymous reviewers for their valuable comments.

References

[1] B. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the*

ACM, 13(7):422–426, July 1970.

[2] L. Fan, P. Cao, J. Aldeida, and A. Broder. Summary Cache: A scalable wide-area Web cache sharing protocol. In *Proceedings of SIGCOMM'98 Conference*, volume 28, pages 254–265, October 1998. Corrected version available at URL: http://www.cs.wisc.edu /~cao/papers/summarycache.html.

[3] D. Johnson, C. Perkins, and J. Arkko. Mobility Support in IPv6. Internet draft, draft-ietf-mobileip-ipv6-19.txt, Work in Progress., October 2002.

[4] T. Narten, E. Nordmark, and W. Simpson. Neighbor Discovery for IP Version 6 (IPv6). RFC 2461, December 1998.

[5] H. Wei and R. D. Gitlin. IP Paging in Mobile Multihop Networks. In *Mobicom 2002 Poster Session*, Atlanta, Georgia, September 2002.

[6] X. Zhang, J. Castellanos, and A. Campbell. Design and Performance of Mobile IP Paging. *ACM Mobile Networks and Applications (MONET)*, 7(2), March 2002.