Evaluation of a Cooperative Caching Scheme for Grid Ad Hoc Networks

Francisco J. González-Cañete, Eduardo Casilari
Department of Electronic Technology
University of Málaga
Málaga, Spain
{fgc, ecasilari}@uma.es

***Abstract*—In this paper, we evaluate the performance of the CLIR (Cross-Layer Interception and Redirection) cooperative caching scheme for ad hoc networks. Although this caching scheme was developed for Mobile Ad Hoc Networks (MANET) we study the application of this kind of algorithms in static grid ad hoc networks. By means of simulations, we evaluate the mean traffic generated in the wireless network, the delay perceived by the users and the percentage of failed searches as a function of the mean time between requests, the Time To Live (TTL) of the documents, the traffic pattern and the cache sizes. We compare the performance of our proposal with another five cooperative caching schemes as well as the option of no using a caching scheme. The simulation results show that our proposal outperforms the other caching schemes in terms of the studied parameters.**

Keywords-cooperative caching; grid; ad hoc network.

I. INTRODUCTION

The aim of a caching scheme is to reduce the traffic generated in the network, as well as the delay perceived by the users and the servers' load [1]. The reduction of the traffic in a wireless network also decreases the probability of collisions and interferences, and hence, the probability of packet loss. Reducing the delay perceived by the users when they request documents improves the user experience and makes the network more attractive to be used. Finally, as a consequence of the caching mechanism, the document requests can be served by other nodes in the wireless network instead of the servers. In a very loaded network, the servers could be a bottleneck as all the requests are sent to them. The caching mechanism mitigates this effect by moderating the overload of the servers so they can reply more requests.

Although many cooperative caching schemes have been proposed for MANETs (Mobile Ad Hoc NETworks) [2], they have not been evaluated for static ad hoc network, that is, wireless networks where the nodes do not move (which may be the typical case of many networking applications such as the sensor networks). The objective of this work is to evaluate the performance of different caching schemes proposed for MANETs in a static grid network.

The rest of this document is organized as follows. In Section II, the related work about cooperative caching schemes for MANETs is presented. In Section III, the proposed caching scheme is described. Section IV defines the system model and shows the performance evaluation of the caching schemes. Finally, Section IV enumerates the main conclusions of this work.

II. RELATED WORK

The cooperative caching schemes for ad hoc networks can be classified into four groups: broadcast-based, information-based, role-based and direct-request. The broadcast-based caching schemes employ broadcast messages as the first choice in order to find the documents in the network. These broadcast messages can be sent to the entire network, as in the case of MobEye [3]. Other schemes such as SimpleSearch [4], follow a more restrictive approach that limits the distance of the messages to four hops. ModifiedSS [5] is an evolution of SimpleSearch that employs GPS (Global Positioning System) in order to send the requests to the direction where the servers are located. Similarly, the caching scheme proposed by Moriya in [6] sends the broadcast messages to the neighbourhood so that, if the document is not found, the request is transmitted to the server.

The information-based cooperative caching schemes employ information of the location of the documents in the network. Nodes obtain this information by analysing the messages that they forward. As examples of this category of caching schemes we can mention: DGA (Distributed Greedy Algorithm) [7], Wang [8], Cho [9] and POACH (POware Aware Caching Heuristic) [10].

Under a role-based caching scheme, each node in the wireless network has a predefined role. That is, they can be caching nodes, requesting nodes, coordinator nodes, gateway nodes, etc. The role-based caching schemes are usually applied to cluster networks. CC (Cluster Cooperative) [11] and Denko [12] are examples of this kind of caching policy.

Finally, the direct-request caching schemes directly send the requests to the server with the hope of being served by an intermediate node in the route from the requester to the server. The proposal by Gianuzzy in [13] is an example of this kind of caching schemes.

However, the groups in this classification of caching schemes are not mutually exclusive. Thus, the caching schemes COOP [14], ORION (Optimized Routing Independent Overlay Network) [15], IXP/DPIP (IndeX

Push/Data Pull/Index Push) [16] and COCA (COoperative CAching) [17] are schemes that employ network information and broadcast requests. On the other hand, COACS (Cooperative and Adaptive Caching System) [18] and GROCOCA (GROup-based COoperative CAching) [19] are role-based caching schemes that also utilize information obtained from the network. In addition, CacheData, CachePath, HybridCache [20] and GroupCaching [21] are direct-request caching schemes that also employ the location information. Finally, ZC (Zone Cooperative caching) [22] and Sailhan [23] use direct requests and broadcast requests depending on some heuristic.

The CLIR cooperative caching scheme was proposed in [24]. It can be classified as a direct-request and information-based cooperative caching scheme. The main novelty of CLIR is the implementation of a cross-layer interception cache technique as well as the optimization of the redirection technique. Its performance was evaluated for MANETs and compared to other five cooperative caching schemes. The objective of this paper is to study the performance of CLIR in a static grid ad hoc network and compare this performance with other caching schemes.

III. PROPOSED CACHING SCHEME

CLIR implements a local cache in every node in the network. This local cache is managed using the LRU (Least Recently Used) replacement policy. Using this cache, every node stores the received documents. Therefore, further requests to the same document will be resolved by the local cache. This is called a local cache hit. As the requests must be forwarded hop by hop from the requester node to the server node, the intermediate nodes in the route from the source to the destination of the requests can reply directly if the requested document is stored in their local cache. This is called an interception cache hit.

When the route from the source node of the request to the destination node has not been created, CLIR utilizes the routing protocol to piggy-back the request in the routing protocol messages. By using this technique, the routing protocol is able to create the route to the destination node and search for the requested document at the same time. If any node that receives the route request message has a copy of the requested document in its local cache, it will reply using the route reply message informing that this node has a copy of the document. When the requester node receives the route reply message, the route between both nodes is created and the requester node will forward the request to the node that has the copy of the document. This is called a cross-layer interception hit. This mechanism allows finding the documents in the network even if the server is not temporarily available.

CLIR also implements a redirection cache that stores information about where the documents are located in the network. This information is obtained from the messages that are forwarded by the mobile node. The redirection cache manages information about the source of the requests and the corresponding replies. It also stores the number of hops and the TTL of the documents and it estimates the time that the documents are stored in the local caches. The redirection cache is managed by means of two LRU lists, one for the documents whose TTL is known and the other with the documents with an unknown TTL. When a node receives a request and the redirection cache contains information of a node that is closer to the original destination of the request, the request is forwarded to this closer node. When the redirected node receives the request, it replies with the document. This is called a redirection cache hit. In the case that the redirected node has evicted the document from its local cache, a redirection error message is sent to the redirection node in order to update the information of the redirection cache.

Finally, CLIR also implements the storage of the replied document in the node located in the middle of the route from the source and destination of the reply. So, the documents can be easily disseminated along the network. In order to avoid the excessive replication of documents, this mechanism is performed if the distance between both nodes is greater than four hops.

IV. PERFORMANCE EVALUATION

In order to evaluate the performance of the proposed cooperative caching scheme we have implemented CLIR using the NS-2.33 [25] network simulator. Additionally, for comparison purposes, the cooperative caching schemes MobEye, HybridCache, COOP, DPIP and SimpleSearch have also been implemented. Each point represented in the figures shown in this paper corresponds to the mean performance evaluation of five simulations using the same parameters but changing the seed. Depending on the simulation, the analysed variable is changed while the rest of the parameters are set to a default value. All figures include a confidence interval of 95% for each performance parameter.

A. Simulation model

Table 1 summarizes the main simulation parameters. We suppose that the nodes in the ad hoc network do not move. Depending on the evaluated configuration, nodes form a regular grid of 5x5, 7x7 or 9x9 nodes. Moreover, the nodes located in the corners of the simulation area, that is, in the positions (x,y)=(0,0) and (x,y)=(1000,1000), are considered to behave as Data Servers (*DS*). For simulation simplicity, we have considered a numeric identification for each document although the caching scheme can be extended to manage URLs. In order to distribute the traffic along the network, the documents with even identification are located in one server while the documents with odd identification are stored in the other *DS*.

Every node that is not a server is programmed to generate requests to the servers during the simulation time. When a request is served, another request is generated after a waiting time period. If the request is not served after a predefined timeout, the request is sent again. The document request pattern follows a Zipf-like distribution that has been demonstrated to properly characterize the popularity of the documents in the Internet [26]. The Zipf law asserts that the probability *P(i)* for the *i*-th most popular document to be requested is inversely proportional to its popularity ranking as shown in (1).

$$P(i) = \frac{\beta}{i^{\alpha}} . \tag{1}$$

The parameter α is the slope of the log/log representation of the number of references to the documents as a function of its popularity rank (*i*).

TABLE 1. SIMULATION PARAMETERS

Parameter	**Default values**	**Other utilized values**
Simulation area (square meters)	1000x1000	
Number of nodes	49	25-49-81
Number of Servers	2	
Number of documents	1000	
Document size (bytes)	1000	
Timeout (s)	3	
TTL (s)	2000	250-500-1000-2000-∞
Mean time between requests (s)	25	5-10-25-50
Traffic pattern (Zipf slope)	0.8	0.4-0.6-0.8-1.0
Replacement policy	LRU	
Local Cache size (number of documents)	35	5-10-35-50
Redirection Cache size (number of registers)	35	
Simulation time (s)	20000	
Warm-up period (s)	4000	
Coverage radio (meters)	250	

As the coverage radio of the nodes is 250 meters and the simulation area is 1000x1000 m^2, the connectivity among neighbour nodes is different for each evaluated grid configuration. Figure 1 shows the connectivity for the 5x5, 7x7 and 9x9 grid configurations. As it can be observed, as the density of nodes increases the number of neighbour nodes grows.

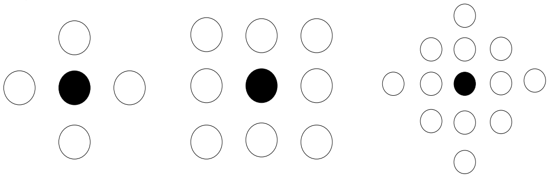


Figure 1. One hop connectivity of a node for 5x5, 7x7 and 9x9 grids.

As performance metrics we consider:

- Traffic load: It measures the mean amount of traffic generated or forwarded by each node during the simulation. As the wireless medium is limited, the greater the generated traffic the greater the probability of interferences and collisions.
- Delay: It is defined as the mean time that a request requires to be served, that is to say, the mean time that a user will have to wait to receive the requested document.
- Timeouts: This metric defines the percentage of requests that have failed and have been requested again because the document has not been received before the timeout.

The figures presented in this section correspond to the evaluation of a 7x7 grid network as the results obtained with the 5x5 and 9x9 networks are very similar. The performance evaluation will be studied as a function of the time between requests, the TTL of the documents, the Zipf slope and the local cache size.

B. *Time between requests*

Figure 2a represents the mean processed traffic by each node as a function of the time between requests. CLIR, DPIP and HybridCache are the caching schemes that generate the lowest traffic, followed by No Cache and SimpleSearch. MobEye generates more traffic because of the use of broadcast messages.

Figure 2b compares the mean delay of the requests and replies. CLIR is the caching scheme with the lowest delay. In fact, it is the only scheme that obtains a lower delay than the option of not using caches. SimpleSearch and MobEye employ a four request-reply messages method, and hence, they experience a greater delay and a greater traffic generation as previously observed. COOP has not been shown in this figure due to the high delay obtained. This behaviour is caused by the timeout needed to perform the direct request to the *DS* after the broadcast request has failed. DPIP also achieves a high delay due to the *DPIP_Timer* parameter that fixes a lower bound to the messages delay. Finally, HybridCache achieves a low performance for high loaded networks although this performance is improved as the traffic load is decreased. This fact is due to redirection loops caused by a wrong redirection management. When time between requests increases, the information stored in the redirection table is obsolete related to the documents stored in the local caches as they are evicted from the local caches before the information can be considered obsolete. As the number of evictions in the local caches decreases the redirection cache is able to obtain more redirection hits because it only takes into account the TTL of the documents to delete the information of the redirection cache.

Figure 2c shows the mean percentage of timeouts per node. HybridCache obtains a high percentage of timeouts due to the bad redirection management as previously explained. Similarly, COOP presents the same behaviour as HybridCache because of the same reasons. Finally, the rest of the caching schemes obtain a percentage of timeouts close to zero. In fact, this should be the normal behaviour of the caching schemes as the servers are always available and it is always possible to create a route to them.

C. *TTL of the documents*

Figure 3a represents the mean traffic processed by each node as a function of the mean TTL of the documents.

CLIR, DPIP and COOP generate less traffic than no Caching for all the studied TTLs. HybridCache is very sensitive to the TTL of the documents and, as the TTL is increased, the generated traffic also soars. This behaviour is due to the redirection cache, which only takes into account this parameter to delete the information in the redirection cache. Consequently, if a node evicts a document from its local cache, the nodes with information about the location of this document in their redirection caches will maintain incorrect data.

Figure 3b compares the mean delay as a function of the mean TTL of the documents. CLIR is the caching scheme that obtains the lowest delay. HybridCache, as shown in the previous study, is very sensitive to the TTL and the delay is highly increased as the TTL is incremented. The rest of the caching schemes obtain delays greater than the case of no Caching due to the four messages needed to obtain the document.

Figure 3c shows the evolution of the percentage of timeouts as a function of the TTL of the documents. COOP and HybridCache are the caching schemes with a percentage of timeouts greater than zero due to the previously commented reason. In fact, the percentage of timeouts is highly increased in HybridCache for TTLs greater than 2000 seconds.

D. *Zipf slope*

Figure 4a depicts the mean traffic processed by node as a function of the Zipf slope. CLIR is the caching scheme that obtains the lowest delay for all the slopes while MobEye and SimpleSearch generate more traffic than the No Caching option due to the broadcast requests. On the other hand, HybridCache also generates more traffic than the No Caching scheme for low slopes. This behavior is due to the replacement policy implemented by HybridCache, called SxO (Size x Order). This replacement policy is very sensitive to the popularity of the documents. Consequently, a low Zipf slope causes the reduction of the local cache hits, increasing the traffic generated in the network.

Figure 4b compares the mean delay as a function of the Zipf slope. The delay obtained by COOP is not shown because it is much greater than the rest of the caching schemes. Only CLIR and HybridCache (for a slope of 1.0) obtain a lower delay than the No Caching scheme. DPIP has a delay of even three times greater than CLIR although this difference is reduced as the Zipf slope increases. CLIR is the caching scheme with the lowest delay for all the considered Zipf slopes.

Figure 4c shows the mean percentage of timeouts per node as a function of the Zipf slope. As observed in previous studies, only HybridCache, COOP and MobEye present a percentage of timeouts different to zero. The behaviour of HybridCache and COOP is due to the incorrect implemented redirection technique. Nevertheless, the percentage of timeouts of these caching schemes is decremented as the Zipf slope increases because, as the Zipf slope increases, the percentage of local and remote cache hits increases and the documents can be served before the timeout. The rest of caching schemes obtain a percentage of timeouts close to zero.

E. *Cache size*

Figure 5a depicts the mean processed traffic by the nodes as a function of the local cache size. As the cache size rises the generated traffic is decreased because the probability of a local cache hit is increased. CLIR, DPIP and COOP are the caching schemes that generate a traffic lower than the No Caching scheme for all the studied cache sizes. MobEye is the caching scheme that generates more traffic due to the use of broadcast requests. On the other hand, HybridCache only performs better than No Caching when the cache size is greater than 20 documents. Hence, HybridCache does not work correctly when using small caches due to the implemented SxO replacement policy.

Figure 5b compares the mean delay as a function of the local cache size. CLIR is the caching scheme with the lowest delay and, in this case, is the one that performs better than the No Caching scheme for all the studied cache sizes. HybridCache presents a big delay for small caches, although it is drastically reduced as the cache size increases. In addition, SimpleSearch and MobEye always obtain a bigger delay than the No Caching scheme for all the studied cache sizes due to the four messages needed to obtain a document. Finally, DPIP shows a delay close to 150 milliseconds due to the limit imposed by the *DPIP_Timer*.

Figure 5c presents the mean percentage of timeouts as a function of the local cache size. As observed in previous studies, only HybridCache, MobEye and COOP show a percentage of timeouts different to zero. This percentage is reduced, especially in HybridCache, as the cache size increases because the probability of local and remote cache also augments.

V. CONCLUSIONS

In this paper, we have evaluated the performance of the CLIR caching scheme applied to static grid ad hoc networks. This evaluation has been performed using the metrics: mean traffic processed by the node, the delay perceived to obtain the requested documents and the percentage of mean timeouts. We have evaluated the influence of the traffic load in the network, the TTL of the documents, the traffic pattern (Zipf slope) and the local cache size. In addition, we have compared the performance of CLIR to the caching schemes HybridCache, COOP, DPIP, SimpleSearch and MobEye. Finally, the performance of CLIR has also been compared to the performance of an ad hoc network that does not implement any caching scheme.

From the set of developed simulations we can conclude that MobEye, COOP and HybridCache are not suitable for static ad hoc networks. We base this assumption in the fact that they obtain a mean percentage of timeouts different to zero. This behaviour is not acceptable in this kind of networks where the servers are always available because the wireless nodes do not move. Taking into account the rest of caching schemes (DPIP, SimpleSearch and CLIR), CLIR always obtains the lowest traffic generation as well as the lowest delay for all the studied situations. In addition, CLIR always presents a better performance than the No Caching Scheme for all the studied parameters and, hence, we can assert that it is suitable for this kind of networks.

ACKNOWLEDGEMENTS

We would like to thank Adela Isabel Fernandez-Anta for revising the syntax and grammar of this article. This study was partially supported by the National Project No. TEC2009-13763-C02-01.

REFERENCES

[1] D. Wessels, Web Caching: Reducing Network Traffic. O'Reilly, 2001.

[2] P. Kuppusamy, K. Thirunavukkarasu, B.Kalaavathi, "A Review of Cooperative Caching Strategies in Mobile Ad Hoc Networks", International Journal of Computer Applications, vol. 29, no. 11, 2011, pp. 22-26

[3] G. Dodero and V. Gianuzzi, "Saving Energy and Reducing Latency in MANET File Access", Proc. 26th International Conference on Distributed Computing Systems Workshops (ICDCSW'06), 2006, pp. 16-20.

[4] S. Lim, W.C. Lee, G. Cao and C.R. Das, "A novel caching scheme for improving Internet-based mobile ad hoc networks performance", Ad Hoc Networks, vol. 4, no. 2, 2006, pp. 225-239.

[5] S. Lim, W.C. Lee, G. Cao and C.R. Das, "Cache invalidation strategies for Internet-based mobile ad hoc networks", Computer Communications, vol. 30, no. 8, 2007, pp. 1854-1869.

[6] T. Moriya and H. Aida, "Cache Data Access System in Ad Hoc Networks", Proc. 57th IEEE Semiannual Vehicular Technology Conference (VTC 2003), April 2006, vol. 2, pp. 1228-1232.

[7] B. Tang, H. Gupta and S.R. Das, "Benefit-Based Data Caching in Ad Hoc Networks", IEEE Transactions on Mobile Computing, vol. 7, no. 3, 2008, pp. 289-304.

[8] Y.H. Wang, J. Chen, C.F. Chao and C.C. Chuang, "A Distributed Data Caching Framework for Mobile Ad Hoc Networks", Proc. 2006 International conference on Wireless communications and mobile computing, 2006, pp. 1357-1362.

[9] J. Cho, S. Oh, J. Kim, K.H. Lee and J. Lee, "Neighbor Caching in Multi-Hop Wireless Ad Hoc Networks", IEEE Communications Letters, vol. 7, no. 11, 2003, pp. 525-527.

[10] P. Nuggehalli, V. Srinivasan and C.F. Chiasserini, "Energy-Efficient Caching Strategies in Ad Hoc Wireless Networks", Proc. 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003), June 2003, pp. 25-34.

[11] N. Chand, R.C. Joshi and M. Misra, "Cooperative Caching in Mobile Ad Hoc Networks Based on Clusters", International Journal on Wireless Personal Communications, no. 43, 2007, pp. 41-63.

[12] M.K. Denko, "Cooperative Data Caching and Prefetching in Wireless Ad Hoc Networks", International Journal of Business Data Communications and Networking, vol. 3, no. 1, 2007, pp. 1-15.

[13] V. Gianuzzi, "File Distribution and Caching in MANET", Technical Report DISI-TR-03-03, DISI Tech University of Genova (Italy), 2003.

[14] Y. Du and S. Gupta, "COOP – A cooperative caching service in MANETs", Proc. Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services (ICAS-ICNS 2005), October 2005, pp. 58-63.

[15] A. Klemm, C. Lindemann and P.D. Waldhorst, "A Special-Purpose Peer-to-Peer Sharing System for Mobile Ad Hoc Networks", Proc. IEEE Semiannual Vehicular Technology Conference (VTC 2003), October 2003, pp. 2758-2763.

[16] G. Chiu and C. Young, "Exploiting In-Zone Broadcast for Cache Sharing in Mobile Ad Hoc Networks", IEEE Transactions on Mobile Computing, vol. 8, no. 3, 2009, pp 384-397.

[17] C.Y. Chow, H.V. Leong and A. Chan, "Peer-to-Peer Cooperative Caching in Mobile Environments", Proc. 24th International Conference on Distributed Computing Systems Workshops (ICDCSW'04), March 2004, pp. 528-533.

[18] H. Artail, H. Safa, K. Mershad, Z. Abou-Atme and N. Sulieman, "COACS: A Cooperative and Adaptive Caching Systems for MANETs", IEEE Transactions on Mobile Computing, vol. 7, no. 8, 2008, pp. 961-977.

[19] C.Y. Chow, H.V. Leong, A. Chan, Group-based Cooperative Cache Management for Mobile Clients in a Mobile Environment, Proceedings of the 33rd International Conference on Parallel Processing (ICPP'04), 2004, pp. 83-90.

[20] L. Yin and G. Cao, "Supporting Cooperative Caching in Ad Hoc Networks", IEEE Transaction on Mobile Computing, vol. 5, no. 1, 2006, pp. 77- 89.

[21] Y. Ting and Y. Chang, "A Novel Cooperative Caching Scheme for Wireless Ad Hoc Networks: GroupCaching", Proc. International Conference on Networking, Architecture and Storage (NAS 2007), 2007, pp. 62-68.

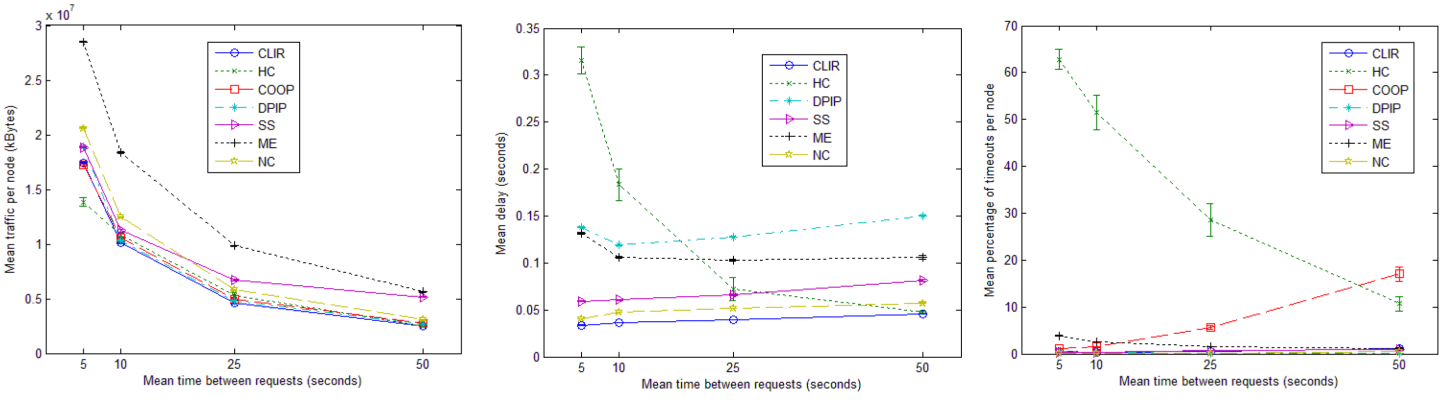
[22] N. Chand, R.C. Joshi and M. Misra, "Efficient Cooperative Caching in Ad Hoc Networks", Proc. 1st International Conference on Communication System Software and Middleware (Comsware'06), January 2006.

[23] F. Sailhan and V. Issarny, "Cooperative Caching in ad hoc Networks", Proc. 4th ACM International Conference on Mobile Data Management (MDM'2003), January 2003, pp. 13-28.

[24] F.J. González-Cañete, E. Casilari and A. Triviño-Cabrera, "A cross layer interception and redirection cooperative caching scheme for MANETs", EURASIP Journal on Wireless Communications and Networking 2012, 2012:63, doi:10.1186/1687-1499-2012-63

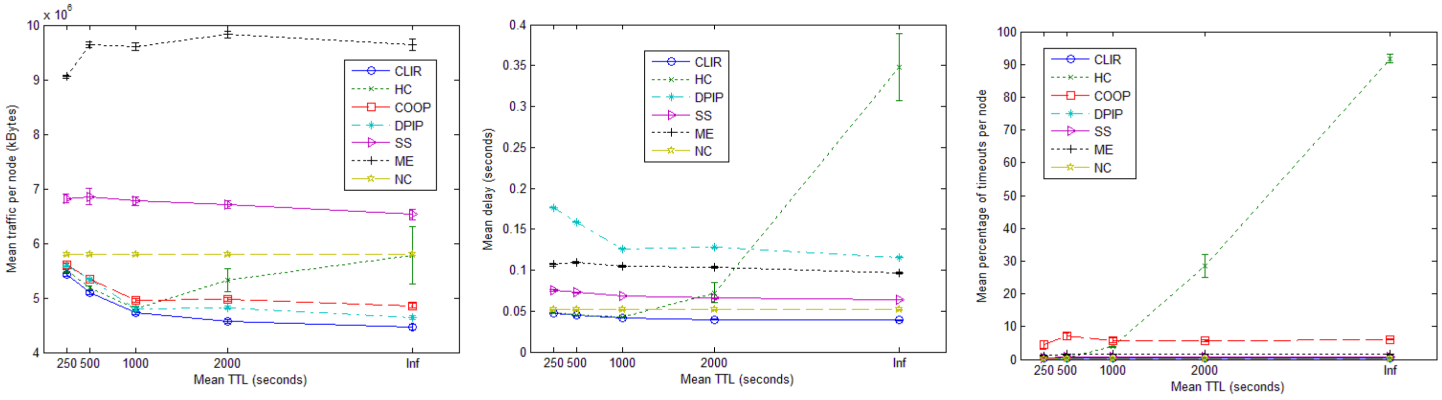
[25] http://www.isi.edu/nsnam/ns/ [retrieved: July, 2012]

[26] L.A. Adamic and B.A. Huberman, "Zipf's law and the Internet", Glottometrics, vol. 3, 2002, pp. 143-150.



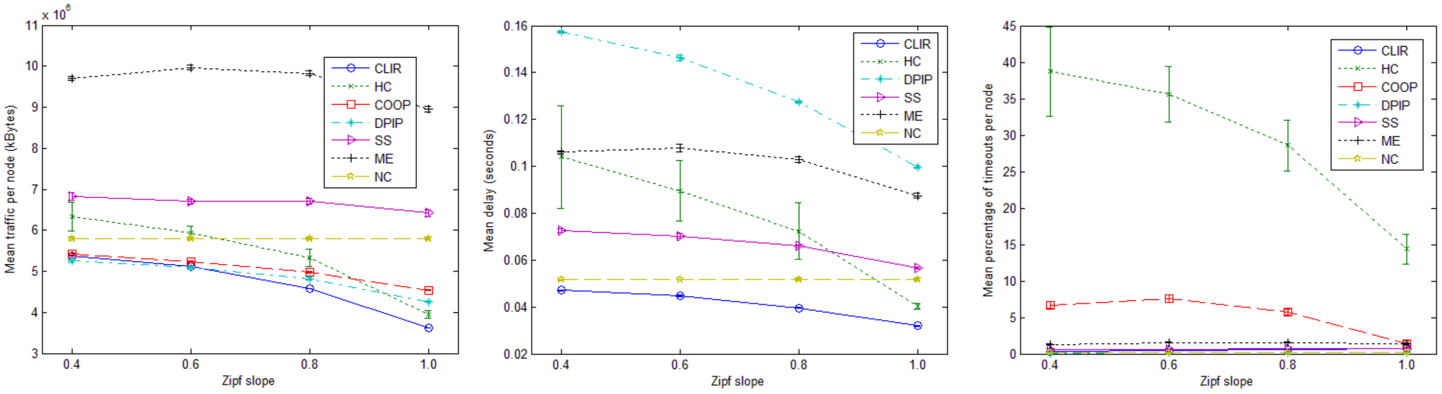
(a) (b) (c)

Figure 2. Mean traffic processed by node (a), delay (b) and percentage of timeouts (c) as a function of the mean time between requests.



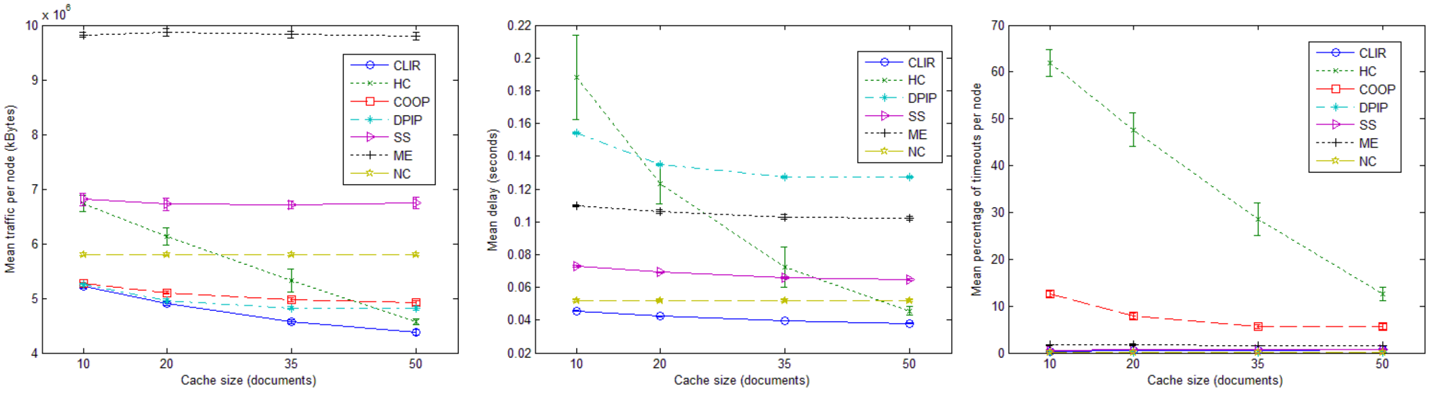
(a) (b) (c)

Figure 3. Mean traffic processed by node (a), delay (b) and percentage of timeouts (c) as a function of the mean TTL of the documents.



(a) (b) (c)

Figure 4. Mean traffic processed by node (a), delay (b) and percentage of timeouts (c) as a function of the Zipf slope.



(a) (b) (c)

Figure 5. Mean traffic processed by node (a), delay (b) and percentage of timeouts (c) as a function of the kcache size.