IWNEST PUBLISHER

Journal of Industrial Engineering Research

(ISSN: 2077-4559)

Journal home page: http://www.iwnest.com/AACE/

Enhancement of the system performance in 802.11 using novel asymmetric cooperative cache

[1]Dr. J. Subash Chandra Bose and [2]M.Murugeswari

[1]*Associate Professor and Head, Department of CSE, Professional Group of Institutions, Coimbatore, India.*
[2]*P.G.Scholoar Department of CSE, Professional Group of Institutions, Coimbatore, India.*

ARTICLE INFO

Article history:
Received 22 February 2015
Accepted 20 March 2015
Available online 23 April 2015

Keywords:

ABSTRACT

To improve the system performance in wireless P2P networks such as ad hoc networks and mesh networks. To design and implementation of cooperative cache in wireless P2P networks, and propose solutions to find the excellent place to cache the data. Utilizing the novel asymmetric cooperative cache approach, where the data requests are transmitted[1] to the cache layer on every node, but the data replies are only transmitted[1] to the cache layer at the intermediate nodes that need to cache the data. This solution not only reduces the suspension of copying data between the user space and the kernel space, it also allows data pipelines to minimize the end-to-end delay.

To Cite This Article: Dr. J. Subash Chandra Bose and M.Murugeswari., Enhancement of the system performance in 802.11 using novel asymmetric cooperative cache. **J. Ind. Eng. Res.,** *1(4), 63-68, 2015*

INTRODUCTION

Enhancement:

In an information technology product, an enhancement[2] is a noteworthy improvement to the product as part of a new version of it..In general, product enhancements include Additional functionality, Error/bug repair and handling Greater processing speed. Better cross-platform compatibility. 802.11 and 802.11x refers to a family of specifications developed by the IEEE for wireless LAN (WLAN) technology. 802.11 enumerate an over-the-air interface between a wireless client and a base station or between two wireless clients. The IEEE accepted the specification in 1997[2]. There are several statements in the 802.11 family: 802.11 ,802.11a,b,e,g,ac,ad,r,X.The co-operative cache can improve the system performance such as ad hoc networks, implementation issues unanswered. The authors are going to study on the effects of different MAC layers and the performance of cooperative will be cache is specified. It is very easy to access file from a nearby live storage node that holds data. So their results show that the asymmetric co-operative cache schema provides the data with less amount of delay time, the asymmetric approach can significantly reduce the data access delay compared to the symmetric approach due to data pipelines.

Relative Work:

Cooperative Caching Schemes:

Below fig 1 illustrates the Cache Path concept. Suppose node N1 seek a data item d_i from N0. When N3 forwards d_i to N1; N3 knows that N1 has a copy of the data. Later, if N2 seek di; N3 knows that the data source N0 is three hops away whereas N1 is only one hop away. Thus, N3 forwards the request to N1 instead[3] of N4.

Many routing algorithms provide the hop count information between the source and destination. Caching the data path for individual data item reduces bandwidth and power consumption because nodes can obtain the data using fewer hops.

Corresponding Author: Dr. J. Subash Chandra Bose, Associate Professor and Head, Department of CSE, Professional Group of Institutions, Coimbatore, India.
E-mail: jsubashme@gmail.com, Phone: +91 9894949054

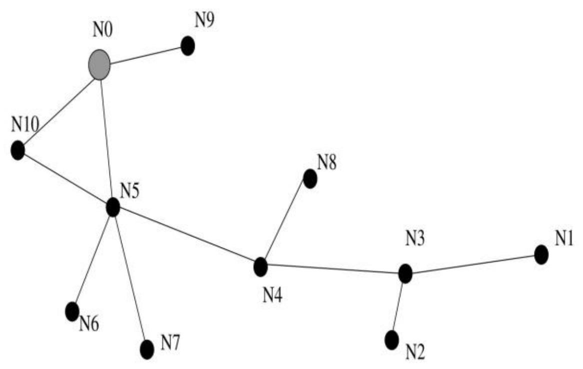


Fig. 1: A wireless P2P network

II. Existing System:

In wireless networks the nodes are unaware whether the other nodes are active are not. When a node wish to transmit some data to[4] other node, a shortest possible path is found by using DSR or AODV routing protocols depending on the network whether it is wireless ad hoc network or wireless mesh network and intermediate nodes along the chosen path need to forward the data from source to destination.

But if any intermediate node along the path is not active, the source node is unaware of this and sends the data. But the data is not received by the destination node[4].

III. Proposed System:

This is proposed to design and implementation of cooperative cache in wireless P2P networks, and to find the excellent place to cache the data. Utilizing the novel asymmetric cooperative cache approach, where the data requests are transmitted[5] to the cache layer on every node, but the data responds are only transmitted to the cache layer at the intermediate nodes that need to cache the data. This solution not only reduces the suspension of copying data between the user space and the kernel space, it also allows data pipelines to minimize the end-to-end delay.

System Architecture:

- The architecture of our cooperative cache middleware, which consists of three parts as shown in fig2.

- Cooperative Cache Supporting Library (CCS)
- Cooperative Cache Daemon (CCD)
- Cooperative Cache Agent (CCA)

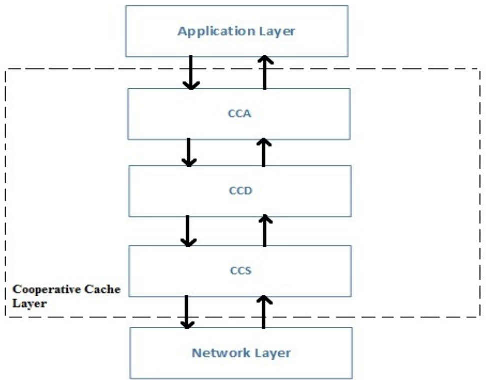


Fig. 2: System Architecture

Cooperative Cache Supporting Library (CCS)

CCS is the core component to provide primitive operations of the cooperative cache, e.g., checking passing by packets, recording data access history, and cache[6] read/Write/replacement primitives[6]. A data cache buffer is maintained at every node to store the cached data items. There is an interface between CCS and the routing daemon, from which CCS obtains the routing distance to a certain node. All these primitive cache functions are enclosed as CCS API to provide a uniform interface to the upper layer.

Cooperative Cache Daemon (CCD)

CCD is the component that implements different cooperative cache mechanisms, namely, Cache Data, Cache Path, and Hybrid Cache. There is one cache daemon for each cooperative cache scheme. It expands the basic CCS API to accomplish the characteristic of each scheme.

Cooperative Cache Agent (CCA)

CCA is the module that maps application protocol messages to corresponding cooperative cache layer messages. It is a connector between CCD and user applications[6]. There is one CCA for each user application.

A greedy cache placement algorithm:

To get the optimal cache placement, the data server needs to compute the aggregate delay for every possible cache placement set. Since there are 2^n (n is the length of the forwarding path) possible ways to choose cache placement set, it takes ω (2^n), which is too expensive. Therefore, we propose a greedy heuristic[1] to efficiently compute the optimal cache placement. Let P*(k) be the optimal cache placement for forwarding

path when only the nearest k hops from the data server are considered as possible cache nodes. With the same condition, let L*(k) be the aggregate delay of the optimal placement P*(k), and p*(k) be the hop distance of the farthest cache node from the data server in P*(k).

When k = 0, no cache node is between the data server and the client, and then the data server N_0 transmits[1] the data directly to the client N_n without reassembling at any intermediate node. All future requests from Ni need to get[1] data from the data server. Therefore, P*(0) =0, p*(k) = 0, and

$$\boldsymbol{L}^*(0) = d_{0,n}(S_D)+\sum_{i=1}^{n-1} f_i .(d_{o,i}(S_D)+d_{0,i}(S_R)) . \Delta t.$$

Given L*(k), P*(k), and p*(k), we check whether to select the intermediate node N_{k+1} as a cache node. If N_{k+1} are selected, we have

$$L(k+1)=L^*(K)+h(S_D)+(d_{p^*(k),k+1}(S_D)+d_{k+1,n}(S_D) \\ -d_{p^*(k),n}(S_D))-\sum_{i=k+1}^{n-1}(d_{p^*(k),i}(S_D)+d_{p^*(k),i}(S_R) \\ -d_{k+1,i}(S_D)-d_{k+1,i}(S_R))f_i\Delta t.$$

Notations:

- *P*: cache placement set.
- *L*: aggregate delay.
- *Pos*: hop distance of the farthest cache node from the data server.
- *f[i]*: Excluded data access frequency on N_i.
- d_D[i,j] : delay of forwarding the data item from N_i to N_j.
- d_R[i.,j] : delay of forwarding the data request from N_j to N_i.
- S_D : size of the data item.
- S_R : size of the data request.
- *R*[*i*] : link throughput between nodes N_i and N_{i+1}.
- *T*[*i,j*] : link throughput between nodes N_i and N_j.
- *L*[*i*] : channel used for the link between nodes N_i and N_{i+1}.
- *h*(S) : cache processing cost for the data size of *S*.
- Δt : the expiration time of the data item.

Cache Placement Algorithm:

```
1:  input: $f[]$, $R[]$, $l[]$, $S_R$, $S_D$.
2:  output: $P$.
3:  for $i$=0 to $n$-1 do
4:  $T[i,i+1]=R[i]$;
5:  end for
6:  for $i$=0 to $n$ do
7:  for $j=i+1$ to $n$ do
8:  compute $T[i,j]$ , $d_D$[i,j], $d_S$[i,j]  using Equ. 4 and 5;
9:  end for
10: end for
11: $L \leftarrow d_D[0,n]$;
12: $P \leftarrow \theta$;
13: $pos \leftarrow 0$;
```

```
14: for $i$=0 to $n$-1 do
15:     $L \leftarrow L + f[i]( d_D[0,i]+ d_R[0,j]) \Delta t$ ;
16: end for
17: for $k$=0 to $n$-$2$ do
18:     $\Delta L_c \leftarrow h(S_D)+ d_D[pos,k+1]+ d_D[k+1,n]- d_D[pos,n]$;
19:     $\Delta L_f \leftarrow ()$;
20:     for $i=k+1$ to $n$-1 do
21:         $\Delta L_f \leftarrow \Delta L_f + d_D[pos,i]+ d_R[pos,i]+ d_D[k+1,i]- d_R[k+1,i] f[i] \Delta t$;
22:     end for
23:     $L' \leftarrow L+\Delta L_c-\Delta L_f$;
24:     if $L'<L$ then
25:         $L \leftarrow L'$;
26:         $P \leftarrow P \cup N_{k+1}$;
27:         $pos \leftarrow k+1$;
28:     end if
29: end for
```

L (k+1) is smaller than L*(k), Nk_{+1} is selected as the cache node since it reduces the aggregate delay. The optimality is updated: L*(k+1) = L(k+1),P*(k +1)= P*(k) [N_{k+1}, and p*(k +1) = N_{k+1}; otherwise, P*(k + 1),L*(k+1), and p*(k +1) keep unchanged from P*(k +1); L*(k+1)and p*(k +1) . Fig. 2.3 shows the detail of the algorithm. The algorithm has complexity of 2^n.

Layered Design:

Cooperative cache is designed as a middleware lying right below the application layer and on top of the network layer (including the transport layer). There are two options for the layered design as shown in fig 3(a),& 3(b). One naïve solution uses cross-layer information[7], where the application passes data request (search key) to the routing layer, which can be used to match the local cached[7] data.

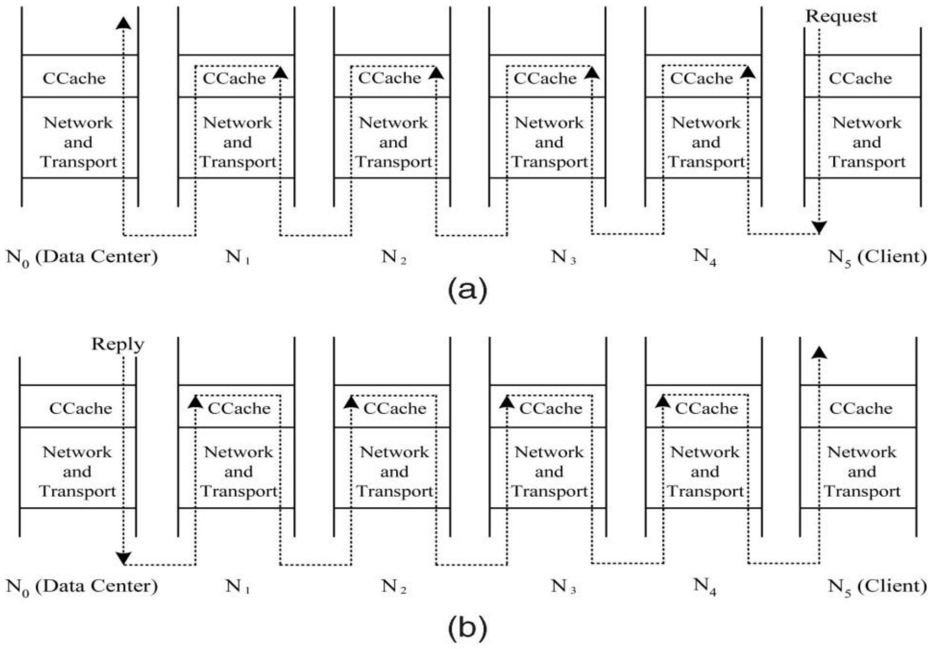
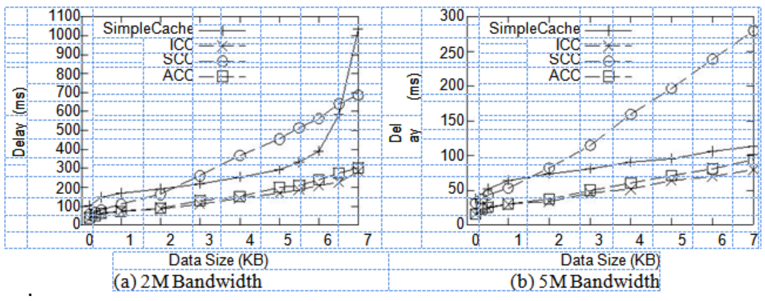


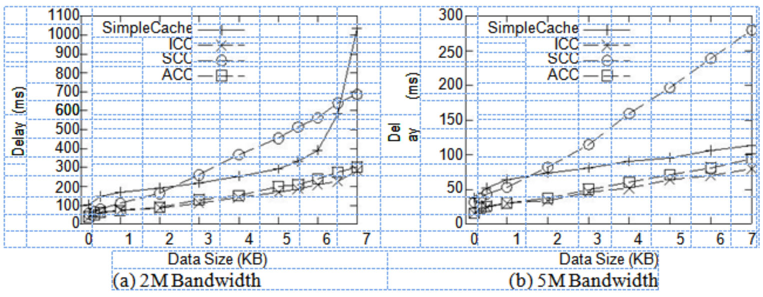
Fig. 3: Layered design. (a) THE REQUEST PACKET FLOW (b) THE REPLY PACKET FLOW

III. Experimental Results:

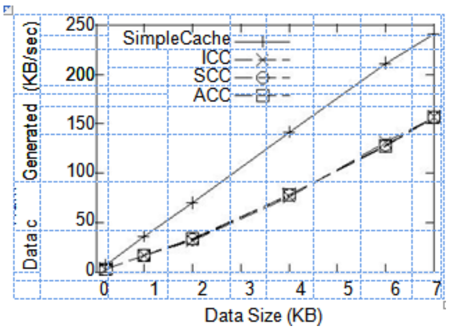
The data access delay comparison in 802.11 networks:



Comparison of the data access delay in wireless mesh networks:



Comparison of the data traffic generated in 5M mesh networks:



Conclusion:

I implemented cooperative cache in wireless peer to peer networks. I even proposed solution to find best place to perform caching. I examined my design for a large-scale network through simulations. An asymmetric approach is proposed. In this, only few nodes are chosen to cache the status of other nodes in its vicinity. All nodes can refer this caching node while[8] sending data to others. The results show that the asymmetric approach outperforms the symmetric approach in traditional 802.11- based ad hoc networks by removing most[8] of the processing overhead.

Future Enhancement:

In the future enhancements, I can choose to find another node which maintains a backup of cache table maintained by Cache cluster heads. This node can take over responsibility of the Cache cluster head in case of its failure. Due to this, the system can be excluded from hanging.

REFERENCES

[1] Zhao, J., P. Zhang, and G. Cao, 2008. “On Cooperative Caching in Wireless P2P Networks,” Proc. IEEE Int’l Conf. Distributed Computing Systems (ICDCS), pp: 731-739.
[2] IEEE, 1999.Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Spec, IEEE 802.11 Standard.
[3] Yin, L. and G. Cao, 2006. “Supporting Cooperative Caching in Ad Hoc Networks,” IEEE Trans. Mobile Computing, 5(1): 77-89.
[4] Perkins, C., E. Belding-Royer and I. Chakeres, 2003. “Ad Hoc on Demand Distance Vector (AODV) Routing,” IETF Internet Draft, draft-perkins-manet-aodvbis-00.txt.

[5] Cao, G., L. Yin and C. Das, 2004. "Cooperative Cache-Based Data Access in Ad Hoc Networks," Computer, 37(2): 32-39.
[6] Bicket, J., D. Aguayo, S. Biswas and R. Morris, 2005. "Architecture and Evaluation of an Unplanned 802.11b Mesh Network," Proc. ACM MobiCom.
[7] Lau, W., M. Kumar and S. Venkatesh, 2002. "A Cooperative Cache Architecture in Supporting Caching Multimedia Objects in MANETs," Proc. Fifth Int'l Workshop Wireless Mobile Multimedia.
[8] http://searchnetworking.techtarget.com/definition/peer-to-peer