An Optimal Cache Partition Based Algorithm For Content Distribution And Replication

***Bandari Muniswamy*[1], *Dr.N.Geethanjali*[2]**
[1]Research scholar, Department of computer Science & Technology,
Sri Krishnadevaraya University,Ananthapuramu,Andhra Pradesh,India.
akhilme2009@gmail.com
[2]Associate Professor,Department of computer Science & Technology,
Sri Krishnadevaraya University,Ananthapuramu,Andhra Pradesh,India.
geethanjali.sku@gmail.com

Abstract*:* ***Generally, users can cast up two sorts of requests, such as elastic requests that contain no delay constraints, and inelastic requests that take an inflexible delay constraint. The distribution of WSN's in multiple areas like, target tracking in battle fields, environmental control needs an optimization for communication among the detectors to serve information in shorter latency and with minimal energy consumption. Cooperative data caching emerged as a productive technique to accomplish these ends simultaneously. The execution of protocols for that network depends mainly on the selection of the sensors which will call for special roles in accordance to the procedure of caching and take forwarding decisions. A perception of Wireless content distribution was shown in which there are numerous cellular base stations, each of which encompass a cache for storing of content. Content is typically partitioned into two disjoint sets of inelastic as well as elastic content. Cooperative caching is shown to be capable to reduce content provisioning cost which heavily depends on service and pricing dependencies among several stakeholders including content providers, web service providers, and end consumers. Hither, a practical network, service, and economic pricing models which are then utilized for making an optimal cooperative caching strategy based on social community abstraction in wireless nets are broken. Inelastic requests are provided by means of broadcast transmissions and here we develop algorithms in support of content spread by means of elastic and inelastic requests. The developed framework includes optimal caching algorithms, analytical models, for evaluating the operation of the suggested scheme. The primary donations are: i) formulation of economic cost-reward flow models among the WNET stakeholders, ii) developing optimal distributed cooperative caching algorithms, iii) characterizing the impacts of network, user and object dynamics, and finally iv) investigating the impacts of user noncooperation,***

Keywords: *Content caching, Elastic request, Inelastic request, Content distribution, Scheduling.*

1. Introduction

For the past few years the growth of wireless content access suggests the requirement for positioning and evolution at wireless stations. In the main content may comprise streaming applications in which file chunks have to be picked up under tough delay constraints [1]. Usually users make two sorts of requests, that is elastic requests that let in no delay constraints, as well as inelastic requests that include a tough delay constraint. Elastic requests are stored up within a request queue at every front end, by each request occupying an exacting queue. In favor of inelastic requests, we take over representation in which users request chunks of content that takes a hard deadline, and prayer is missed if time limit cannot be met. The objective of elastic requests is to fix queue, in an attempt to contain finite delays. The objective of inelastic requests is to pair up a convinced target delivery ratio of the entire requests that have to be determined to make sure smooth play out. Whenever each time an inelastic request is sent packing, updating of deficit queue is executed by means of quantity that is proportional to delivery ratio. ratio. We would like average value of the deficit to be zero [2]. In our work we build up algorithms for content distribution with elastic as well as inelastic requests. We build use of a request queue to completely find out popularity of elastic capacity. Correspondingly, deficit queue determines essential service for inelastic requests. Content might be refreshed at regular intervals at caches.

In cooperative caching, a node not only can access to the substance stored in its local cache, but it can also search its desired item within stored data items in other caches. The fact that people in the same location tend to partake in common interests supports the idea of cooperation among their mobile devices. For instance, people in a classroom share similar interests on the topic of class or people in a conference have similar interests along the topic of presentation. Under the premise of having similar interests by users collocated in the same arena, it's quite possible for a node to get its desired data item in other nodes' caches. We divide users into different clusters, with the estimate that all users in each cluster are geographically close such that they have statistically similar channel conditions and are capable to access the same base stations. Notice that multiple clusters could be present in the same mobile phone based on the dissimilarity of their channel conditions to different base stations. The bandwidth available at a BS (Base Station) [4-6] s ε S is reflected using a bandwidth-cost function, Cs (k) (r), which stands for the cost incurred by us for serving requests at a total rate of r. An example, bandwidth-cost function can be of the mannikin:

$$C_s^{(k)}(r) = \exp\left\{a\left(\frac{r-R_s^{(k)}}{k}\right)\right\}$$

In the conventional download model, a user downloads, data items directly from a Content Provider's (CP) server over a Communication Service Provider's (CSP) network. With the local caching a user first searches its local cache before downloading the content from the content provider. All the same, in cooperative caching mechanism, an alternative content access approach would be to look for the local MSWNET for the requested content after local search for the requested content fails and before downloading it from the CP's server. Therefore, mobile users, depending on their respective locations, will be potentially related to several BSs from which content may be immediately downloaded. In this context, BSs can be heterogeneous in terms of bandwidth and storage capabilities. For example, a nearby farm to call BS will provide more bandwidth than the heavily loaded cellular BS; however, the probability of obtaining a desired content at the them to cell is lower due to its lower cache capacity.

The following constraints affect system performance

(i) The wireless network between the caches to the users has a finite capacity,

(ii) Each cache can only host a finite quantity of substance, and refreshing content in the hoards from the media vault incurs a price.

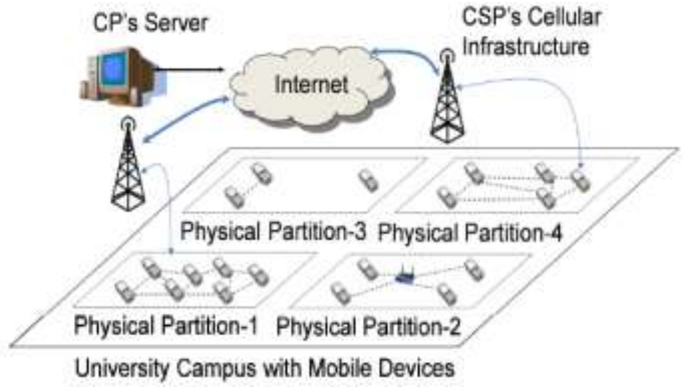

Fig.1 Partition overview of network in wireless content distribution

In order to deploy a cooperative caching framework, each node must support few functionalities. For instance, due to limited storage capacity in each mobile device, a replacement mechanism is required to fit a new downloaded data item when a cache is full. Furthermore, every caching node should manage content of its cache based on the other nodes' needs [7–11]. To boot, a cache resolution mechanism is also needed to get a requested data item among the remote caches. Cache resolution addresses how to resolve an object request either by seeing the object in the local cache or remote caches in the mesh partition. Later on a request is started from a user (i.e. An application on a mobile device), the device first performs a local search within its local cache. If it fails to obtain the requested object, a web search is executed for the requested item within the current divider. If this measure also fails, and the device has access to the Internet, the requested item will be downloaded straight from the content provider's server. A node effectively cooperates with all nodes in its ring search. When a guest receives a search request for one of its locally cached objects, it sends a unicast ACK to the petitioner. And so the requester starts downloading the data item from the responding client.

2. Related work

Cache management refers to policies that control object placement in the caches and determine objects distributed in the partition. Cache management comprises admission and replacement policies. Admission policy: A node adjusts its cooperative behavior depending on the state of other nodes' caches within its ring search. A simple cooperation policy for a node is not to store an object if it has already been stored inside the division. Under this policy, the total figure of different items stored inside the segmentation can be increased. This in turn can increase the partition item availability in the case no Internet link is usable in the division.

Cache replacement: To store a new downloaded data item when cache is full a node executes a replacement policy in order to determine as to which data item from its cache should be substituted. The topic of cooperative caching has attracted significant attention in the literature concerning various types of broadcast systems; on the Web [15], in file servers [16], and then along. However, the very limited capabilities of the sensor nodes (in terms of energy, memory, and computation), the particularities of the wireless channel (variable content), and the multi-hop style of communication, turns the solutions suggested in the aforementioned environments, of limited usefulness. Cache consistency refers to mechanisms that try to preserve the content of cached data items and original data items the same. This is a vital part of a caching framework, especially when content of data items is subject to frequent modifications. Thither is a broad scope of schemes for maintaining cache consistency. Some of these techniques are specifically planned for mobile ad hoc wireless networks [17]. In general, cache consistency strategies fall into two broad classes. In drag-based systems, a client initiates data validation by polling the content server to determine if data has been modified since it was salted away. In push-based systems, the content server initiates the data invalidation by notifying the caching nodes.

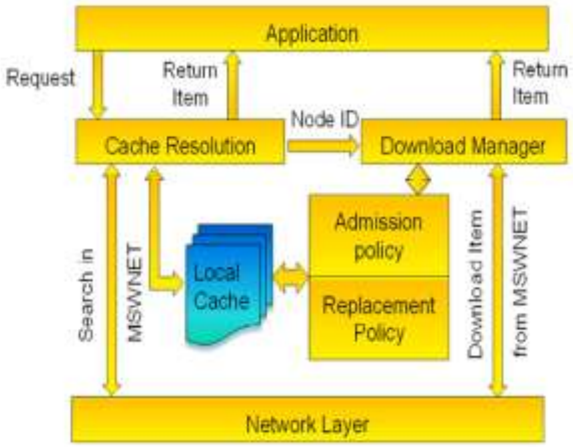

Fig. 2 An architecture of co-operative caching

In distributed systems over wireless networks based on multi-hop communication, cooperative caching has been proven a very effective strategy to reduce the communication latency and maintain energy. Nuggehalli et al. [17] addressed the problem of energy-conscious, cache placement in the wireless ah hoc network, and [18] considered the cache placement problem of minimizing total data access cost in ad hoc networks with multiple data points and nodes with limited storage capacity, and gave a polynomial-time centralized approximation algorithm to attack the problem, since it is NP-hard. Though these works address cache placement issues.

The most important, relevant works are those described in [1], [2], [14]. The work covered in [14] considered cache replacement issues for wireless ad hoc networks, but in the setting of a very special kind of cooperation; a node which requests a data search either in its local cache or in the caches of its 1-hop neighbors (otherwise forwards the petition to a ficticious data center). Therefore, remote hits cannot happen take in this protocol. Yin & Cao proposed the Hybrid cooperative caching protocol, which exploited both data and node locality in a homogeneous style, but this policy was proved inferior to NICoCa, described in [2] which took special consideration to select appropriate "central" nodes to transmit out and coordinate the cooperation. Still, the choice of "central" nodes does not bring into consideration the remaining energy of sensor nodes. Therefore, the energy consumption of significant nodes will give birth as a consequence the reduction of network lifetime and finally the network fragmentation.

Jeffrey E. Wiesel their et al develops the Broadcast Incremental Power Algorithm, and adjusts it to multicast operation as good. This algorithm exploits the broadcast nature of the wireless communication environment, and addresses the demand for energy-efficient performance. We have placed some of the underlying issues associated with energy-efficient broadcasting and multicasting in infrastructure less wireless networks, and we have presented preliminary algorithms for the resolution of this problem. Our surveys demonstrate that improved public presentation can be obtained when exploiting the properties of the wireless medium; i.e., Networking schemes should reflect the node based nature of wireless communications, rather than simply adapt, link-based schemes originally developed for wired networks. In especial, the Broadcast Incremental Power (BIP) Algorithm, which exploits the wireless multicast advantage, provides better performance than the other algorithms we have examined over a broad range of network models. [1].

Meghana M Amble et al objective is to design policies for request routing, content positioning and content eviction with the goal of small user delays. Stable policies ensure the finiteness of the request queues, while good polices also lead to short queue lengths. We first design a throughput-optimal algorithm that solves the routing-placement-eviction problem. The design yields insight into the impact of different cache refresh policies on queue length, and we construct throughput optimal algorithms that engender short queue lengths. We exemplify the potency of our approach through simulations on different CDN topologies.. Future work includes streaming traffic with requests that have hard delay constraints, and which are dropped if such a constraint cannot be met. [2]

Somanath Majhi et al Fourier series based curve fitting with the options of nonlinear least squares method and trust-region algorithm is used to measure limit cycle parameters in the presence of measurement noise. Examples are given to illustrate the value of the proposed method Relay feedback identification in process control can lead to erroneous results if the system parameters are estimated from the approximate describing function approach. Exact analytical expressions are derived and on the basis of these expressions an identification procedure is suggested which is capable of estimating the parameters of a class of process transfer functions. When the limit cycle test measurements are error free, the accurate values of the model parameters are estimated. The solutions for the three examples show the generally accepted point about the DF method that its accuracy increases with the relative order of the plant transfer function which is successively 1, 2, and 5 in the examples. The present method can be applied to identify the class of non minimum phase processes in the presence of measurement noise.[3]

Bo Zhou et al formulate this stochastic optimization problem as an in finite horizon average cost Markov decision process (MDP). It is well-known to be a difficult problem and there generally only exist numerical solutions. By using relative value iteration algorithm and the special structures of the request queue dynamics,. we consider the optimal dynamic multicast scheduling to jointly minimize the average delay, power and fetching costs for cache-enabled content-centric wireless networks. We formulate this stochastic optimization problem as an infinite horizon average cost MDP. We show that the optimal policy has a switch structure in the uniform case anda partial switch structure in the non uniform case. Moreover, in the uniform case with two contents, we show that the switch curve is monotonically non-decreasing. The optimality properties obtained in this paper can provide design insights for multicast scheduling in practical cache-enabled content centric wireless networks [4].

3. Existing system

Different cooperative caching mechanisms have been introduced for web proxies and file organization. These cooperation mechanisms can be broadly categorized to hierarchical, directory-based, hash-table and multicast-based attacks. For example, Harvest is a hierarchical approach in which a user's request is forwarded to a cache hierarchy till the request is set up at some degree. In such a hierarchical approach, no info is interchanged between the caches in different grade. In a directory-based advance, like Summary Cache [22], each cache exchanges a summary vector of its data items to the other tutors. Then, during cache resolution a cache forwards the petition to a cache that has a copy of the requested data item. Squirrel is a fully decentralized, peer-to-peer cooperative web cache, based along the thought of enabling web browsers on desktop machines to share their local caches by using a hash table. There has been several important works done on content caching algorithms, but there were fewer efforts made on the interaction of coaching as well as nets. In our study we study algorithms for content placement as well as scheduling in wireless nets. Converting of caching as well as load balancing problem into one of queuing and scheduling is thus motivating. In our work we consider a arrangement in which inelastic and elastic requests coexist. Our intention was to even out system in conditions of fixed queue lengths in support of elastic traffic and zero average deficit in support of inelastic traffic. The techniques that were applied are based on methods of scheduling. The difficulty of caching, as well as content scheduling has been considered for online web caching as well as distributed storage systems.

In our study we are concerned with solving joint content placement as well as scheduling difficulty for elastic as well as inelastic traffic within wireless systems. In undertaking so, we see the value of anticipating demand for various types of content and the impact it has on aiming of caching algorithms. The content distribution network will be at wireless gateway, which might be a cellular base station all the way through which users achieve network access. It can benefit from the intrinsic broadcast nature of the wireless medium to convince numerous users concurrently [3]. In the given fig. 1 there are numerous cellular base stations (BSs), each of which contains cache to store up content. The cache content at regular intervals can be refreshed all the way through accessing a media vault. We separate users into several clusters, with the proposal that the entire users in each cluster are geographically secure such that they contain statistically related channel conditions and are capable to access similar base stations.

To encourage the end-consumers to participate in this cooperative caching, a content provider pays a rebate to an end-consumer when it provides a data item to another device. A key question for cooperative caching is how to store contents in nodes such that the average content provisioning cost in the network is minimized. In this chapter we show that in a stationary mobile wireless network an optimal caching strategy exists which can minimize the average cost per accessed object. Numerous clusters might be present in similar cell based on difference of their channel conditions towards different base stations. The requests that are made by each cluster are combined at a logical entity that is known as front end connected with that cluster [4]. The front end might be running on any of devices within cluster or else at a base station, and its intention is to carry on track of requests connected with users of that cluster. The limitations that have an effect on system operation are: wireless network among caches towards users containing fixed capacity; each cache can host a fixed amount of content; and refreshing content in caches from media vault gains a cost. The common objective of the existing cooperative caching mechanisms for wireless networks is to achieve high object availability at the partition level. This is achieved by avoiding the storage of duplicated objects within a network partition. While improving partition level availability, these approaches offer low availability at the node level, because only one copy of each popular object exists within a partition.

4. Distribution of content for optimal caching in Wireless systems

While there has been important work on algorithms of content caching, there is much less on interaction of caching as well as networks. Users can build two kinds of requests, that is: elastic requests that contain no delay constraints, and inelastic requests that contain an inflexible delay constraint. In a request queue, elastic queries are stored at each front end, by a request engaging a particular queue and its objective is to balance the queue, in an attempt to enclose finite delays. Intended for inelastic requests, we adopt a model in which users request content chunks that include a strict deadline, and request is dropped if deadline cannot be met.

Let P_L be the probability of finding a requested object in the local cache (i.e. local hit rate), P_V be the probability that a requested object can be found in the network (i.e. Remote hit rate) after its local search failed, and P_M be the probability that a requested object is not found in the network. We can write P_M in terms of P_V and P_L as:

$$P_M = 1- P_L - P_V$$

The proposal here is to fulfill a convinced target delivery ratio. Each time when an inelastic request is dropped, restructuring of a deficit by a quantity that is proportional to delivery ratio. Converting caching and load balancing difficulty into one of queuing and scheduling is thus interesting. We consider a system in which both inelastic as well as elastic requests co-occur. The requests that are made by each group are collected at a logical entity termed as front end that is associated with that cluster. The front end might be running on any of the devices within cluster or at base station, and its function is to continue path of requests that are connected with users of that group. The restrictions that have an effect on system operation are wireless network among caches to users containing fixed capacity; each cache hosting only a fixed amount of content; refreshing content in caches from media vault incurring a cost [5].

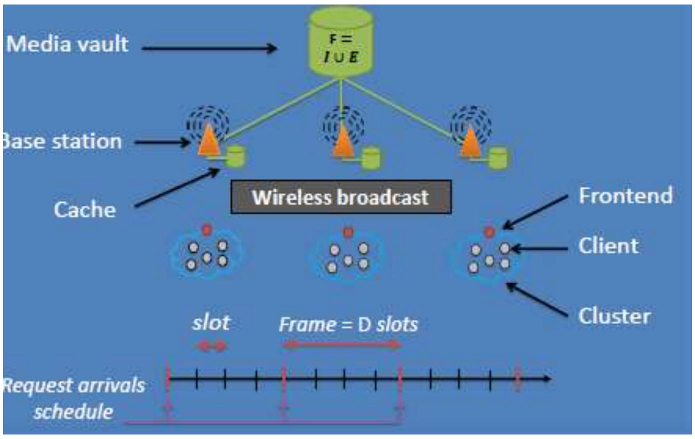

Fig 3: An overview of content distribution.

Our purpose was to improve system regarding finite queue lengths in support of elastic traffic and zero average deficit value in favor of inelastic traffic. A natural location towards placing caches intended for a content distribution network would be at wireless gateway, which may possibly be a cellular base station by which users get hold of network access. The base stations make use of numerous access schemes and consequently each base station can maintain multiple instantaneous uncast transmissions, in addition to a single broadcast transmission. It is moreover likely to learn other scenarios by means of our framework.

5. An overview of proposed Algorithms for content Distribution

In our work we study algorithms for content placement as well as scheduling in wireless networks. Here the objective is to make available placement as well as scheduling algorithms that can execute any set of severely practical requests. By means of capability to predict requests, we might potentially decide on elastic content distribution system a priori which is similar to find suitable joint distribution of content placement and service schedule. The solution would give way to a set of caching as well as scheduling choices, and a possibility with which to utilize each one on basis of channel realizations [5]. It was observed that prediction of elastic requests has restricted value in circumstance of devising suitable algorithms of content distribution. Elastic requests are supposed to be served all the way through uncast communications among caches as well as front ends, while the base stations broadcast inelastic contents to inelastic users. High node level availability is desirable so that the popular objects are available to nodes even when they are completely isolated without being connected to any other node. We consider the following premises:

- In that location are N clients in the population. Clients are indistinguishable and act independently of one another.
- The total number of documents is *n*. For ease, we model documents as indivisible, rather than as a compound, and assume that accesses to objects are autonomous.
- The fraction of all requests that are for the *i*-th most popular document, or the “popularity” of this document, is denoted by p_i. We accept that documents follows a Zipf-like distribution [15], i.e., that pi is proportional to $1/i^{\alpha}$ for some constant α. The significant feature of a Zipf-like distribution is that it is hard-tailed – a substantial fraction of the probability mass is condensed in the hind end, which in this case signifies that a substantial fraction of requests go to the relatively unpopular documents. As α increases, the distribution becomes less heavy-tailed, and a larger fraction of the probability mass concentrates on the most popular documents.
- The distribution of time between requests made by the client population is exponential with parameter λN, where λ is the average client request rate.
- The distribution of time between changes to a document is exponential with parameter μ, independent of document size and latency, but not independent of popularity. We use two separate document change distributions, one for popular documents with mean μ_p and another for unpopular documents with mean μ_u. The turn of popular documents is n_p. Document change can be applied to model either expiration or actual change.
- The chance that a requested document is cacheable is p_c.
- The average document size is *E(S)*. Document size is independent of document popularity, latency, and pace of alteration.
- The last-byte latency to the host that houses that document has average value *E (L)*. Last-byte latency is independent of document popularity and document rate of alteration.

Parameter	Value
α	0.8
λ	590 req/day(appr)
p_c	0.6
μ_p	1/14days(slow) - 1/5min(fast)
μ_u	1/186 days - 1/85 days
E(S)	7.7 KB
E(L)	1.9 s

Fig 4 Table for the simulation Parameters.

A limitation of the existing mechanisms is their inability to provide high availability at both partition and node levels. The main objective is to develop cooperative caching mechanisms that can improve object availability within WNET partitions as well as at individual nodes. This is achieved by letting each node in a partition to store a set of objects while allowing certain level of duplication in the partition.

The simulation parameters are as follows:

Number of ECs in a static partition (*V*)	: 40
Download cost (C_d)	: 10
Rebate-to-download-cost ratio (β)	: 0≤β≤1
Cache size in each mobile device (*C*)	: 50
Zipf parameter (α)	: 0.8
Object population (*N*)	: 5000

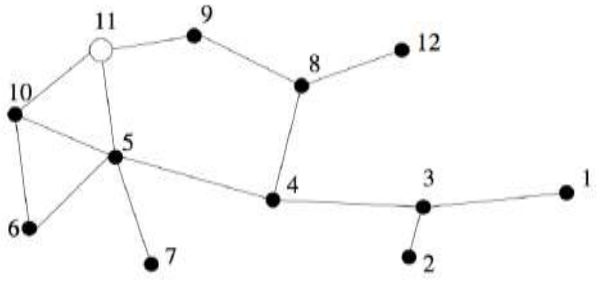

Fig.5 Overview of cache path

The second scheme is Cache Path, where a node caches the path of previously served objects. For example, if node 1 requested an object from node 11, then node 3 remembers that the object is cached in node 1 and in the time to come, if node 2 requests the same object from node 11, then node 3 will redirect the request to node 1. To give the greedy approach to this problem, we will schedule jobs successively, while ensuring that Noah picked job overlaps with those previously scheduled. The central design factor is to decide the social club in which we consider jobs. There are several ways you can do so. Say for instance, that we pick jobs in increasing order of size.

Consider any solution *S* with at least *k* jobs.

- We claim by induction on *k* that the greedy algorithm schedules at least *k* jobs and that the first *k* jobs in the greedy schedule finish, no later than the first *k* jobs in the chosen solution.
- This immediately implies the outcome because it demonstrates that the greedy algorithm schedules at least as many jobs as the optimal answer. We at once examine the title.
- The base case, $k = 0$ is trivial. For the inductive step, consider the $(k + 1)$th job, say *J*, in the solution *S*. Then this job begins after the k^{th} job in S ends, which happens after the k^{th} job in the greedy schedule ends (by the induction hypothesis).

Thus, it is possible to augment the greedy schedule with the job *J* without introducing any conflicts. The greedy algorithm finds a candidate to augment its solution and in particular, picks one that finishes no later than the fourth dimension at which *J* ends. This finishes the proof of the claim.

We define two graph structures g_{Left} and g_{Right} in line 3 since each DN(Delegate Node) will bipartite the received graph *G* into two partitions (sub-graphs) and selects one of the member nodes to act as a *DN* for each partition; hence, we define two node structures dn_{Left} and dn_{Right} in line 3. The two string variables s_{Left} and s_{Right} (line 4) are used to keep track of the already selected *DN*s. Alg. 1 is triggered by the CM(Content Manager) with *G* representing the whole graph and the string variable *s* initialized with the NULL string. The CM selects itself to act as a DN for the network graph *G* and initializes both s_{Left} and s_{Right} to the value of *s* (line 6, which is a NULL string when the instance is called by the ContentManager(CM)

Algorithm:

Alg. 1 Input: Network graph G.

Output: A string representation of the DBT returned to the CM.

```
1. Function BTree(Graph g; String s)
2  Graph  g_Left; g_Right
3  Node  dn_Left; dn_Right
```

```
String $s_{Left}; s_{Right}$
$s_{Left}$   $s_{Right}$   $s$
if |g| = 2 then
$dn_{Left}$   this
$dn_{Right}$  {g} \ n dnLeft
return $dn_{Left}$ || {g}
end
else if |g| = 3 then
$dn_{Left}$   rand(g) ₡ s
$g_{Right}$   {g}\ $dn_{Left}$
$dn_{Right}$   rand($g_{Right}$) ₡  s
forward ($g_{Right}$; s) ! $dn_{Right}$
wait ($dn_{Right}$)
return this || $dn_{Left}$ || $dn_{Right}$
end
```

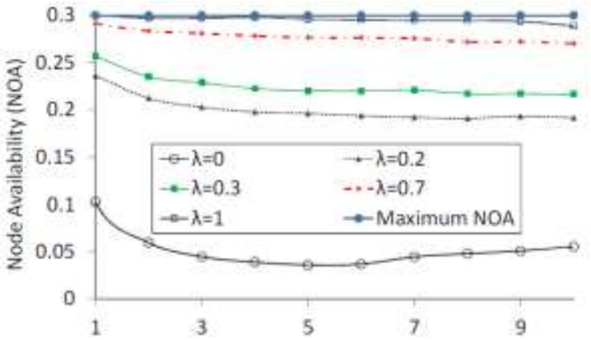

Fig:6 Graph for the simulation results.

To control the level of duplication, the cache space in a node is divided into two separate areas. The first area is dedicated for storing the most popular (duplicated across all nodes in the partition) objects to guarantee high availability within a completely isolated node.

	Packets	Received (Conventional)		Received(Proposed)		Threshold
λ=0	121	0.9	0.85	0.3	0.11	70.3
	579	0.87	0.95	0.29	0.26	52.4
λ=1	539	0.96	0.98	0.17	0.16	41.2
	352	0.89	0.92	0.22	0.088	21.3

And the second area is dedicated to store partition-wide unique objects to achieve high level of availability at the partition level. Consequently, planning for cache placement of inelastic content cannot be achieved in advance thus even predicting necessary quantity of cache resources is not simple thus, we conclude that prediction of arrival rates in support of inelastic traffic is of marginal value. The contribution of the proposed cache partitioning mechanism is to provide high levels of node and partition availabilities, and at the same time to reduce the generated network traffic.

6. Conclusion

Networks of content distribution are positioned at wireless gateway, which might be a cellular base station all the way through which users achieve network access. It can advantage from basic broadcast nature of wireless medium to convince numerous users concurrently. In our work we construct algorithms for content distribution by means of elastic as well as inelastic requests and consider a system where inelastic and elastic requests coexist. This is achieved by using a novel cache partitioning method which can be used for fine grain object duplication control within isolated network partitions. It was demonstrated that by using this cache partitioning strategy, the proposed mechanism is able to outperform the existing schemes while reducing network traffic. The above conclusion holds for both stationary and mobile networks. Balancing of the system in terms of fixed queue lengths in support of elastic traffic and zero average deficits in support of inelastic traffic was the intention of our work. The procedures that were employed are based on methods of scheduling and in our work we are concerned in solving joint content placement as well as scheduling difficulty for elastic as well as inelastic traffic within wireless systems. In designing these schemes, we showed that knowledge of the arrival process is of limited value to taking content placement decisions.

References

[1] N. Abedini and S. Shakkottai, “Content caching and scheduling in wireless broadcast networks with elastic and inelastic traffic,” in *Proc.IEEE WiOpt*, 2011, pp. 125–132.

[2] I. Hou, V. Borkar, and P. Kumar, “A theory of QoS for wireless,”in *Proc. IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr. 2009, pp.486–494.

[3] R. M. P. Raghavan, *Randomized Algorithms*. NewYork,NY,USA: Cambridge Univ. Press, 1995.

[4] P. Cao and S. Irani, “Cost-awareWWWproxy caching algorithms,” in*Proc. USENIX Symp. Internet Technol. Syst.*, Berkeley, CA, Dec. 1997,p. 18.

DOI: 10.18535/Ijecs/v5i4.23

[5] K. Psounis and B. Prabhakar, "Efficient randomized Web-cache replacement schemes using samples from past eviction times,"*IEEE/ACM Trans. Netw.*, vol. 10, no. 4, pp. 441–455, Aug. 2002.

[6] N. Laoutaris, O.T. Orestis, V.Zissimopoulos, and I. Stavrakakis, "Distributed selfish replication," *IEEE Trans. Parallel Distrib. Syst.*, vol.17, no. 12, pp. 1401–1413, Dec. 2006.

[7] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *Proc. IEEE INFOCOM*, San Diego,CA, USA, Mar. 2010, pp. 1–9.

[8] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems an scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no.12, pp. 1936–1948, Dec. 1992.

[9] X. Lin and N. Shroff, "Joint rate control and scheduling in multihop wireless networks," in *Proc. 43rd IEEE CDC*, Paradise Islands, Bahamas,Dec. 2004, vol. 2, pp. 1484–1489.

[10] A. Stolyar, "Maximizing queueing network utility subject to stability:Greedy primal-dual algorithm," *Queueing Syst. Theory Appl.*, vol. 50,no. 4, pp. 401–457, 2005.

[11] A. Eryilmaz and R. Srikant, "Joint congestion control, routing, and mac for stability and fairness in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1514–1524, Aug. 2006.

[12] J. Jaramillo and R. Srikant, "Optimal scheduling for fair resource allocation in ad hoc networks with elastic and inelastic traffic," in *IEEE INFOCOM 2010*, San Diego, CA, March 2010.

[13] M. M. Amble, P. Parag, S. Shakkottai, and L. Ying, "Content-Aware Caching and Traffic Management in Content Distribution Networks," in *to appear in IEEE INFOCOM 2011*, Shanghai, China, April 2011.

[14] M. Neely, "Energy optimal control for time-varying wireless networks," *Information Theory, IEEE Transactions on*, vol. 52, no. 7, pp. 2915–2934, 2006.

[15] F. Foster, "On the stochastic matrices associated with certain queueing processes," *Ann. Math. Statist*, vol. 24, pp. 355–360, 1953.

[16] M. Neely, "Energy optimal control for time varying wireless networks," *IEEE Trans. Information Theory*, vol. 52, no. 2, pp. 2915–2934, July 2006.