Dynamic Content Placement for Mobile Content Distribution Networks

Wagner M. Aioffi, Geraldo R. Mateus, Jussara M. Almeida, and Raquel C. Melo

Department of Computer Science
Federal University of Minas Gerais - Brazil
{aioffi,mateus,jussara,raquelcm}@dcc.ufmg.br

Abstract. As wireless networks increase in popularity, the development of efficient content distribution techniques to meet the growing and constantly changing client demand becomes a necessity. Previous content distribution network proposals, targeting mainly wired networks, are not adequate to handle the high temporal and spatial variability in client demand caused by user mobility. This paper proposes and analyzes a wireless dynamic content distribution network framework that replicates content in response to changes in client demand in order to reduce the total traffic over the network backbone. We propose a mathematical programming model to determine the offline optimal solution as well as an online algorithm based on demand forecasting. Using a previously developed mobility simulator, we show our new online algorithm outperforms the traditional centralized and distributed static placement approaches, reducing the total traffic over the network backbone in up to 78%, compared to the best previous approach analyzed.

Keywords: Dynamic content placement, mobile networks, demand forecasting.

1 Introduction

As the number of wireless network users increases and with the advent of high bandwidth third-generation mobile networks, it becomes necessary to develop new system design techniques to meet the increasing demand. Next generation wireless telecommunication systems should be designed as a high-capacity system, able to cope with the envisaged overwhelming traffic volume.

A Content Distribution Network, or simply CDN, replicates popular content at a number of servers, placed closer to high demand sites, in order to reduce network bandwidth requirements and latency observed by the end user. CDNs have been shown to be effective for managing content distribution to a large number of users in wired networks. In those cases, the assumption of a static client demand distribution is usually made [3,13,7,10,15].

In wireless networks, client demand is expected to vary significantly not only over time but also over space, due to user mobility. For example, a city downtown area may have a higher demand for business content during working hours. However, one might expect the demand to drop significantly as people return to their homes after work. Thus, the design of a content distribution system for wireless networks must rely on dynamic replica placement strategies in order to cope with the possibly high variability of client demand.

To the best of our knowledge, content replication for variable client demand has only been studied recently. In [3,5], the authors address the problem for specific network topologies. A heuristic for dynamic replication of Internet applications in more general topologies was recently proposed in [14]. Optimization models targeting system availability [18] and for providing lower bounds on the cost-effectiveness of different replica placement heuristics [8] have also been proposed. Nevertheless, these studies propose and evaluate solutions for wired networks.

This work proposes and analyzes a Wireless Dynamic Content Distribution Network Model (WDCDNM). In WDCDNM, the CDN should be dynamically reconfigured in response to, possibly high, temporal and spatial variations in client demand with the goal of reducing total traffic over the network backbone. Reconfiguring the CDN implies that new replicas of currently popular objects should be added to the network. Similarly, previously created replicas of objects that have just become unpopular should be removed from the network.

We formulate a mathematical programming model, based on well-known combinatorial optimization models, which represents the offline optimal version of the problem. We also propose an online heuristic algorithm, which uses demand forecasting to reconfigure the network for future demand, to approximate the optimal solution given by the mathematical model. Since the offline model requires *a priori* knowledge of all future demand, it cannot be implemented in a real system. However, it offers an ideal lower-bound that can be used to analyze the competitive efficiency of the online algorithm [1].

An extensive performance analysis of our online algorithm, using a previously developed mobility simulator [11], showed that it produces solutions with a total network traffic that is only up to four times the offline optimal. Furthermore, compared to existing centralized and distributed static content placement approaches, our heuristic saves up to 78% of network traffic over the best static approach analyzed.

This paper makes the following contributions:

– An offline optimization model that provides a lower bound for the total traffic over the network backbone for Wireless Content Distribution Networks. The offline model can be implement total and partial content replication.
– A novel and efficient online algorithm for dynamic content placement in wireless networks that reduces network traffic in up to 78%, compared to static content placement.

The remaining of this paper is organized as follows. Section 2 discusses related work on CDNs. Section 3 describes the mathematical programming model and the online heuristic algorithm. Section 4 reports a performance evaluation of our solutions. Conclusions and future work are offered in Section 5.

2 Content Distribution Networks

A Content Distribution Network (CDN) is a system that allocates replicas of its contents over servers geographically scattered, placed, typically, close to high demand sites in order to reduce client latency and network bandwidth requirements. Content placement is a key problem in the design of efficient CDNs. Determining the number and location of replicas that should be created for a certain content is a non-trivial problem that depends on the solution of a complex tradeoff. Associated with each new replica is a replication cost which is proportional to the bandwidth required for transferring a copy of the content from the closest server where it is currently stored to the new replica server. There is also an extra cost associated with maintaining the local replica up-to-date. On the other hand, a new replica contributes to reducing the bandwidth required for serving client requests and, thus, reducing client latency.

The content placement problem, previously shown to be NP-Complete [4], has been studied mostly for static client demands. Optimal solutions for specific topologies [9,10] and heuristics for more general topologies [6,13,15] have been proposed. Content placement for dynamic client demand has been studied only very recently [3,5,8,18,14]. In [3], the authors address the problem with the goal of minimizing the number of replicas deployed in a tree topology while meeting client QoS and server capacity constraints. Dynamic content placement heuristics for replicating content inside a cluster-based cache are proposed in [5]. In [8], the authors propose an offline integer programming model for choosing the most cost-effective heuristic for a given system, workload and performance goal. By using specific constraints, the proposed model offers lower bounds for the cost associated with a number of different heuristics. A replication cost optimization models targeting system availability is proposed and formalized in [18]. Finally, Rabinovich *e*t al [14] study the problem in the context of Internet application replication in general topologies. They propose a heuristic, called ACDN, for dynamic replica placement based on the observed past demand that aims at minimizing the network bandwidth requirements. Like the solutions proposed in [3,5,8,18], the ACDN algorithm was proposed and evaluated for wired networks.

CDN design solutions targeting, specifically, wireless networks receive little attention so far. Previous work addresses mainly the server selection problem [16,17]. In [16] the authors address the server selection problem in a streaming media mobile CDN with servers arranged hierarchically, with the goal of reducing the number of mobile client handoffs. In [17], the authors propose a mobile streaming media CDN controlled by SMIL (Synchronized Multimedia Integrated Language) with the goals of improving streaming media quality, supporting client mobility and efficiently utilizing network resources.

This paper addresses the content placement problem in the specific context of wireless networks, where client demand may vary significantly over time and space. It proposes and evaluates not only an online dynamic content placement heuristic but also an offline optimal solution. Unlike the optimization models proposed in [8] and in [18], which target latency and system availability in wired networks, respectively, our offline optimal solution aims at minimizing total traffic over the backbone of wireless content distribution networks.

3 Wireless Dynamic Content Distribution Networks

This section defines our Wireless Dynamic Content Distribution Network Model (WDCDNM), a framework for managing content replication and placement in a wireless network aiming at reducing the total traffic in the network backbone. It also presents the mathematical programming model, which provides an offline optimal solution, and the online heuristic algorithm.

WDCDNM uses the general term “content” to represent a collection of objects (for instance, the collection of objects of a given site). Each content is stored in at least one server in the system, which is referred to as its origin server. Mobile users, or simply clients, request individual objects of a content. A request to a certain object is always sent to the *currently* closest server, due to server wireless coverage limitation. This server may or may not have a replica of the requested content. If it has, the request is served locally, incurring no traffic over the network backbone. Otherwise, it forwards the request to the closest server that has the content replica, and relays the response to the client. In this case, the indirect request service generates traffic over the network backbone between the two servers involved in the operation. Similarly, replication and replica maintenance operations also generate traffic over the backbone. A maintenance operation occurs whenever a content is modified in its origin server.

WDCDNM characterizes each content c by three size parameters: number of bytes transmitted during a replication (sr^c), number of bytes transmitted when a request for an object of the content is indirectly served (sm^c), and number of bytes transmitted during a maintenance operation. Note that WDCDNM assumes all objects within the same content have equal sizes, although different contents may have different total sizes. Extensions for allowing heterogeneous object sizes within the same content are straightforward. Partial and total content replication can be implemented in the WDCDNM framework by simply setting sr^c and si^c appropriately. In case of partial replication, indirect request service and replication are performed on a per-object basis and, thus, for a given content c, $sr^c = si^c$. Otherwise, replication is performed for the whole content at once. Thus, $sr^c \geq si^c$. The value assigned to sm^c depends on whether a maintenance operation is performed by transmitting a new copy of the content or simply “patches” with the changes to be performed on the old replica.

In response to the current spatial distribution of client demand (i.e., number of client requests per unit of time) for objects of different contents, WDCDNM performs content replication and/or removal of previously created replicas in the servers, with the goal of minimizing the total traffic over the network backbone. This traffic is calculated as the sum of the total traffic generated by partial/total content replication operations, the total traffic generated by replica maintenance operations and the total traffic generated by indirected responses to client requests. Each traffic component is estimated as the product of the number of bytes transferred in each operation (given by sr^c, si^c and sm^c), the "distance" (i.e., number of hops, geographical distance, etc) between the two servers involved in the operation and the number of operations performed.

The decision of adding a new content replica to a server or removing an existing one is based on the local demand for the content, and should be revisited periodically to absorb spatial and temporal changes in client demand. The following sections describe our two implementations of the WDCDNM framework: the offline optimal solution and the online (heuristic) algorithm.

3.1 Offline Optimal Solution

The offline optimal solution of WDCDNM is formulated as a mathematical programming model. The model is based on the well-known Uncapacitated Location Problem [12], a classical NP-Complete combinatorial optimization problem, extended to represent the spatial and temporal variations in client demand caused by user mobility. The following notation is used in the model formulation:

C:	Set of contents available in the network;
T:	Set of reconfiguration periods;
S:	Set of servers;
$dist_{i,j}$:	Distance between servers $i \in S$ and $j \in S$;
o^c:	Origin server of content $c \in C$;
d_i^{tc}:	Total demand, expressed as the number of requests sent by local clients to server $i \in S$ for objects of content $c \in C$ during period $t \in T$;
m^{tc}:	Flag that indicates whether content $c \in C$ is modified at its origin server during period $t \in T$;
bi_{ij}^c:	Traffic generated over the backbone if server $j \in S$ indirectly serves a request, originally sent to server $i \in S$, to an object of content $c \in C$;
br_{ij}^c:	Traffic generated over the backbone if server $i \in S$ (totally or partially) replicates content $c \in C$ from server $j \in S$;
bm_i^c:	Traffic generated over the backbone if server $i \in S$ has to update a local replica of content $c \in C$;
y_{ij}^{tc}:	Binary variable that indicates whether server $i \in S$ replicates content $c \in C$ from server $j \in S$ during period $t \in T$;
a_i^{tc}:	Binary variable that indicates whether server $i \in S$ has the content $c \in C$ during period $t \in T$;
x_{ij}^{tc}:	Number of times that server $i \in S$ forwards a request for an object of content $c \in C$ to server $j \in S$ during period $t \in T$.

The problem is then formulated as follows:

$$\min \sum_{t\in T}\sum_{s\in S}\sum_{i\in I}\sum_{j\in I} x_{ij}^{tc} bi_{ij}^{c} +$$
$$\sum_{t\in T}\sum_{s\in S}\sum_{i\in I}\sum_{j\in I} y_{ij}^{ts} br_{ij}^{c} + \sum_{t\in T}\sum_{s\in S}\sum_{i\in I} a_i^{tc} m^{tc} bm_i^{c}$$

Subject to:

$$\begin{aligned}
bi_{ij}^{c} &= si^{c} \times dist_{i,j} && \forall c \in C, \forall i \in S, \forall j \in S && (1)\\
br_{ij}^{c} &= sr^{c} \times dist_{i,j} && \forall c \in C, \forall i \in S, \forall j \in S && (2)\\
bm_{i}^{c} &= sm^{c} \times dist_{io^{c}} && \forall c \in C, \forall i \in S && (3)\\
\sum_{i \in S} a_i^{tc} &\geq 1, && \forall t \in T, \forall c \in C && (4)\\
\sum_{j \in S} x_{ij}^{tc} + d_i^{tc} \cdot a_i^{tc} &= d_i^{tc}, && \forall t \in T, \forall c \in C, \forall i \in S && (5)\\
x_{ij}^{tc} &\leq d_i^{tc} \cdot a_j^{tc}, && \forall t \in T, \forall c \in C, \forall i \in S, \forall j \in S && (6)\\
a_i^{(t+1)c} - a_i^{tc} &\leq \sum_{j \in S} y_{ij}^{tc}, && \forall t \in T, \forall c \in C, \forall i \in S && (7)\\
y_{ij}^{tc} &\leq a_j^{tc}, && \forall t \in T, \forall c \in C, \forall i \in S, \forall j \in S && (8)\\
x_{ij}^{tc} &\geq 0, && \forall t \in T, \forall c \in C, \forall i \in S, \forall j \in S && (9)\\
y_{ij}^{tc} &\in \{0, 1\}, && \forall t \in T, \forall c \in C, \forall i \in S, \forall j \in S && (10)\\
a_i^{tc} &\in \{0, 1\}, && \forall t \in T, \forall c \in C, \forall i \in S && (11)
\end{aligned}$$

The objective function minimizes the total traffic over the network backbone. The first set of summations represents the total traffic generated bye request indirect attendance, the second set of summations represents the total traffic generated by replication operations and, finally, the last set of summations represents the total traffic generated by replica maintenance operations.

Constraints (1), (2), and (3) computes the traffic generated by each indirect request service, replication and maintenance operation, respectively. The set of constraints (4) ensures that there is at least one server for each content on the network. The set of constraints (5) ensures that client requests are served, either locally or by remote servers. The set of constraints (6) ensures that a server only forwards a request to a server containing the requested object. Constraints (7) guarantee that each replication does result in a new content replica. The set of constraints (8) ensures that the content being copied during a replication is stored in the originating server. Constraints (9), (10) and (11) ensure that variables x_{ij}^{tc} are non-negative and variables y_{ij}^{tc} and a_{ij}^{tc} are binary, respectively. For simplicity, the model does not impose constraints on server storage requirements. However, it could be easily extended to include such constraints.

The model defined above implements the offline version of our WDCDNM and can be solved to optimality. Unlike previous static content placement optimization models [9,10,13], it uses the set of parameters d_i^{tc} to model demand variation over time and space, since the spatial location of each mobile user at a given time determines the server to which its requests are sent. Because it needs a priori knowledge of all future demand, the model can not be applied to a real system. Nevertheless, it provides a lower-bound on the total network traffic generated for efficient content placement in mobile networks and thus, can be used to assess the efficiency of our online approximate algorithm, described next.

3.2 Online Algorithm

In order to solve large and practical problems, we propose and evaluate a new online heuristic algorithm for dynamic placement of content in wireless CDNs. The algorithm uses a statistical forecasting method, called Double Exponential Smoothing [2], to predict future demand for different contents on each server, and uses the predictions to decide whether to add a new content replica or to remove an existing one.

The Double Exponential Smoothing algorithm works as follows. Let y_t be the actual demand, in number of client requests, observed during period t at a server. Let α be a smoothing factor. The method calculates the prediction $\hat{y}_{t+\tau}$, made at the beginning of period t for the expected demand for period $t+\tau$ at the server, as follows:

$$\hat{y}_{T+\tau}(T) = (2 + \tfrac{\alpha\tau}{1-\alpha})S_T - (1 + \tfrac{\alpha\tau}{1-\alpha})S_T^{[2]}$$

where:

$$\begin{aligned} S_T &= \alpha y_T + (1-\alpha)S_{T-1} \\ S_T^{[2]} &= \alpha S_T + (1-\alpha)S_{T-1}^{[2]} \end{aligned}$$

The α parameter is used to control how quickly recent demand observations are incorporated in the prediction. In other words, α is the weight given to recent observations in the prediction of future demand.

Our new online heuristic is a distributed greedy algorithm, which is executed simultaneously and independently by each server. At the beginning of each reconfiguration period, each server independently decides whether it should locally replicate any of the contents available in the system and whether it should remove local replicas, based on the predictions of future *local* demand for the following δ periods. More precisely, each server keeps track of the number of local client requests to each content received during each period. It then uses the Double Exponential Smoothing method to predict future local demand for each content. The predictions are then used to estimate the total network traffic that would be generated from replicating the content and maintaining the new replica, or from simply maintaining the replica, if it already exists, for the following δ periods. It also estimates the total network traffic that would be generated during those periods if it had to forward local client requests to the current closest replica. The content is replicated in the server only if the estimated traffic incurred by replication and future maintenance is lower than the estimated traffic incurred by indirect request service. Similarly, the local replica is removed from the server if the estimated maintenance traffic is higher than the estimated traffic caused by indirect request service. In summary, each server independently makes a decision that, at the time, minimizes its share of the total traffic generated for serving its *predicted* future demand. The algorithm is shown in Algorithm 1.

Note that our algorithm takes a different approach from the previously proposed ACDN dynamic content placement algorithm [14]. In particular, there are four key points that distinguish them. First, the ACDN algorithm makes replication decisions based on the demand observed in the past, whereas our algorithm

Algorithm 1 New Online Content Placement Algorithm for Wireless Networks

```
for $\forall$ content $c \in C$ do {Executed by server $i \in S$ at beginning of each period $t \in T$}
  for $k = 1, \cdots, \delta$ do
    $\hat{y}_{t+k} = Forecast\_Local\_Demand(c, i, t, k)$ {Use Double Exponential Smoothing
    Method to forecast local demand for period $t + k$ in the future.}
  end for
  $d^{ic} = \sum_{k=1}^{\delta} \hat{y}_{local_{t+k}}$ {Compute predicted local demand for the next $\delta$ periods.}
  {Let $j$ be the server with the closest replica of content $c$ to server $i$.}
  {Estimate total traffic over the backbone for indirect service in the next $\delta$ periods.}
  $bi^c = d^{ic} \times si^c \times dist_{i,j}$
  {Estimate total traffic over the backbone for replicating content $c$ in server $i$. }
  $br^c = sr^c \times dist_{i,j}$
  {Estimate total traffic over the backbone for maintaining new replica of content $c$
  in server $i$. Flag $m^{t,c}$ indicates whether content $c$ changes during period $t$.}
  $bm^c = \sum_{k=1}^{\delta} m^{t+k,c} \times sm^c \times dist_{i,o^c}$
  if $i$ does not have a replica of content $c$ then
    if $bi^c > (br^c + bm^c)$ then
      Add new replica of $c$ in server $i$
    end if
  else if $i$ has the content $c$ and is not the origin server of $c$ then
    if $bi^c < bm^c$ then
      Remove replica of $c$ from server $i$
    end if
  end if
end for
```

uses predictions of future demand, which in turn, are based on the *evolution* of past demand. Second, the ACDN algorithm adopts a "push" strategy. In other words, at each reconfiguration period, a server that currently has a replica decides as to whether it should send it to other servers from which it received a large number of indirect requests in the past. In contrast, our algorithm follows a "pull" strategy. Third, the "push" approach makes an implementation of the ACDN algorithm in space-constrained servers more difficult. The server that is initiating a content replication operation should know whether the destination server has enough local space to store the new replica. Finally, our online algorithm is more efficient than the ACDN algorithm. Whereas our algorithm has time complexity equals to $O(|C|)$, their algorithm has complexity $O(|S| \times |C|)$, where $|C|$ and $|S|$ are the numbers of contents and servers in the system. A quantitative performance comparison of both algorithms is left for future work.

4 Performance Evaluation

This section evaluates the performance of our new online dynamic content placement algorithm, comparing it with the offline optimal solution and with the previous centralized and distributed static placement approaches. Section 4.1 briefly describes the mobility simulator used to evaluate the algorithms. The most relevant performance results are presented in section 4.2.

4.1 Mobility Simulator

The mobility simulation and demand generation are performed using a mobility simulator developed in [11]. The simulator models a twenty-kilometer radial city, divided in area zones. The division is based on population density and natural limits (e.g., rivers, highways, railway tracks). Taking mobile telecommunication requirements into account, it seems reasonable to assume that each area zone is equal to a network area (e.g., macrocell, local exchange area). The area zones are connected via high-capacity routes, which represent the most frequently selected streets for movement support, and are grouped into four area types: city center, urban, suburban and rural. The simulator also includes a number of content servers with fixed locations. Figure 1 shows a representation of the modeled city with the server locations indicated by small dark circles. The city has 32 area zones (eight per city area types), four peripheral routes (one per area type) and four radial high-capacity routes.

The simulator models residences, workplaces, schools and other points such as shopping centers and parks as movement attraction points, i.e., locations where people spend considerable amounts of time. Figure 2 shows the frequencies of different types of movement attraction points in each city area type.

The user population is divided into four mobile groups according to the mobility characteristics of the individuals and to the demand they generate for different content types. The four user groups are defined as follows: 5% of the users are 24-hour delivery boys, 60% are common workers, 30% are housekeepers and the remaining users are taxi drivers. The typical mobility behavior of each user group is defined in a movement table. In this table, the day is divided into time periods and a probability of a user of the group being at a certain location is associated to each period. Typical movement tables are given in [11]

Each user within a group generates a number of calls during simulation. The per-user call inter-arrival times are exponentially distributed with means 14, 7, 14 and 18 minutes, for 24-hour delivery boys, common workers, housekeepers and taxi drivers, respectively. Once connected, a user issues a number of requests at rate 1 request per second. The number of requests issued by a user during a call depends on the call duration, which is also exponentially distributed with mean 60 seconds, for all groups. Within each user group, relative content popularity follows a Zipf-like distribution (Prob(request content c) $= K/c^{\alpha}$, where $\alpha > 0$ and K is a normalizing constant [19]), with parameter $\alpha = 0.84$.

4.2 Results

We evaluate the performance of the new online dynamic content placement algorithm comparing it to two traditional static placement strategies: centralized and distributed. In the centralized approach, each content is stored in only one server in the network, whereas the distributed approach allows for each content to be replicated in a fixed number of servers. In both cases, the location of the content replicas do not change over time, although the content may itself change, triggering maintenance operations, in case of multiple replicas. We also compare our algorithm with the ideal lower-bound provided by the offline optimal solution.

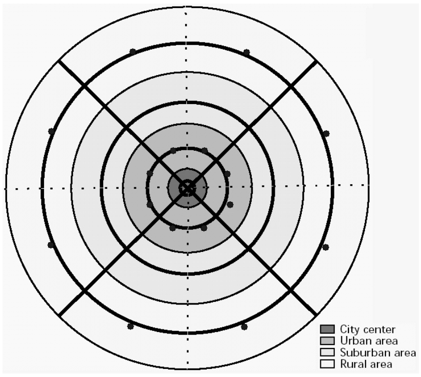

Fig. 1. The Simulation Model: City Areas and Server Locations

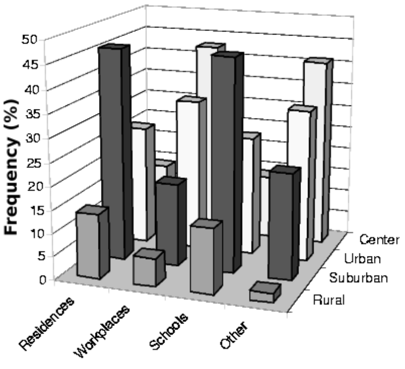

Fig. 2. Frequency of Movement Attraction Points over the City Areas

In our simulations, we experiment with a distributed approach with 2 and 4 replica servers. Furthermore, in both centralized and distributed static approaches, the content replicas are placed in the more central servers (see Figure 1). We make the assumption that the distance between two servers is the linear distance shown in Figure 1. Furthermore, unless otherwise stated, in all experiments, we set the number of contents to 3 and the number of mobile users to 5000. The reconfiguration period is set to 10 minutes and the simulation runs for 90 periods. The α and δ parameters of the demand forecasting method are set to 0.2 and 7, respectively. We also make the conservative assumption that the content modification rate is one per period, for all contents. Thus, maintenance operations must be performed over the deployed content replicas at each period.

We run a large number of experiments varying several system parameters and covering a large design space. The next sections present the most relevant results obtained in our experiments.

Replication, Indirect Service, and Maintenance Content Sizes
This section analyzes the impact of the three content size parameters, namely, the replication size (sr^c), the indirect service size (si^c) and the maintenance size (sm^c), which represent the number of bytes transferred in each specific operation. Since the traffic generated in each operation is proportional to the number of bytes transferred, the relative content sizes impact directly a server decision for replicating a content or removing an existing replica and thus, the performance of our algorithms. In the following experiments, we assume a baseline configuration where $sr^c = si^c = sm^c = 1KByte$, for contents $c \in C$.

We start by evaluating the impact of the replication size on the total network traffic generated over the backbone if the indirect service and maintenance sizes are fixed at the baseline value and the replication size increases to 20, 30, 40 and 50 times the baseline. Figure 3 shows the total traffic generated for the online algorithm and offline optimal solution. For comparison purposes, it also shows the traffic for the centralized and distributed static approaches. Note that the variation in the replication size has no impact on the static approaches. However, as replication size increases, the replication component of the total traffic starts to dominate. In response, the online and offline optimal approaches tend to reduce the number of replications. As expected, the offline optimal solution, having the knowledge of all future demand, makes replication decisions that will lead to significant reductions in the future traffic, especially given the low maintenance cost. On the other hand, the online algorithm uses predictions of future demand for the following δ period and thus, may make some sub-optimal replication decisions which will not pay-off in the long run. Nevertheless, it is interesting to note that in the worst case, for a replication size equal to 50 times the baseline, the online algorithm produce a solution with total network traffic which is only four times larger than the offline optimal. Furthermore, compared to the best static approach (distributed with 4 servers), our online algorithm reduces total network traffic in 78%, for a replication size equal to 20 times the baseline, and in 21% for a replication size equal to 50 times the baseline.

Fixing the replication size at 50 times the baseline and the indirect service size at the baseline, Figure 4 shows the total traffic generated by each algorithm as the maintenance size increases. Recall that we assume a content modification rate of one per period, for every content. Thus, as maintenance size increases, it becomes less cost-effective to maintain a replica in a server with low client demand. In this scenario, total network traffic for the dynamic algorithms increases significantly. However, the performance of the static distributed approaches degrades even more significantly, especially for large number of replica servers, because fixed location replicas must be maintained even during low demand periods.Compared to deploying four fixed-location replica servers, our online algorithm reduces total network traffic in up to 50%. In particular, for maintenance sizes larger than 20 times the baseline, the online dynamic algorithm chooses not to replicate any content and, thus, performs similarly to the centralized approach. Furthermore, our online algorithm results in an increase in the total traffic that is only 14% higher than the offline optimal. If maintenance size is very large, both algorithms decide for not replicating. Thus,the knowledge of future demand does not give a significant advantage to the offline optimal.

Finally, we now fix both replication and maintenance sizes equal to 50 times the baseline, and vary the indirect service size. Figure 5 shows the results. As the content indirect service size increases, it becomes more cost-effective for a server to replicate the content locally than to forward their client requests to the closest replica. Thus, the dynamic algorithms generate significantly better results than the static approaches. In particular, our online algorithm, which produces solutions within 18% of the offline optimal, reduces the total network traffic generated by the best static approach analyzed by up to 61%.

Number of Simultaneous Mobile Users

This section analyzes the impact of the number of simultaneous mobile users simulated on the total network traffic generated over the backbone by our new online and offline optimal solutions and by the traditional static centralized and distributed approaches. We use homogeneous size configurations with all three size parameters equal to 10 KBytes. Figure 6 shows that the dynamic content

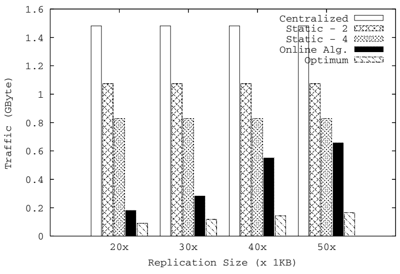

Fig. 3. Total Network Traffic as a Function of Replication Size sr^c. ($si^c = sm^c = 1KB, \forall c \in C$)

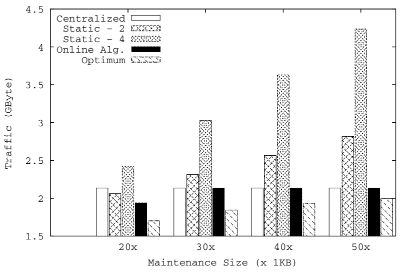

Fig. 4. Total Network Traffic as a Function of Maintenance Size sm^c. ($si^c = 1KB, sr^c = 50KB, \forall c \in C$)

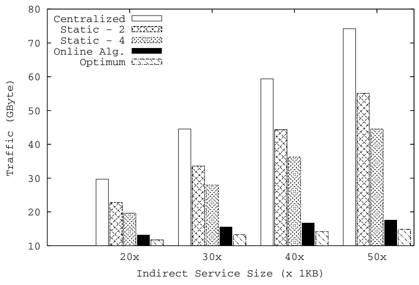

Fig. 5. Total Network Traffic as a Function of Indirect Request Service Size si^c. ($sr^c = sm^c = 50KB, \forall c \in C$)

placement solutions scale much better with the number of concurrent mobile users than the static approaches. With 500 thousand mobile users, the online algorithm reduces total traffic in 75%, over static placement with 4 replicas.

Simulation Time

This section evaluates the increase in the cumulative network traffic as a function of simulation time. As shown in Figure 7, for a configuration with all three size parameters equal to 10 KBytes and 2000 mobile users, the dynamic solutions outperforms the static approaches significantly, as time progresses.

Traffic Overhead Due to Management Operations

Content replication and maintenance operations incur a management traffic overhead, which must be payed off by a reduction in the traffic of indirect client responses in order to be beneficial to the system. Otherwise, total network traffic increases and dynamic content placement may perform worse than static content placement. Figure 8 shows, for each algorithm analyzed, the portion of the total traffic due to management operations, for a configuration with all three size pa-

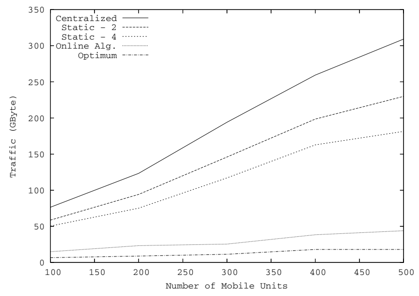

Fig. 6. Total Network Traffic as a Function of the Number of Users

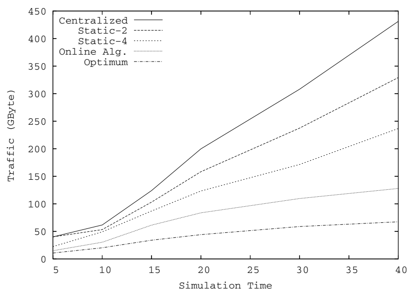

Fig. 7. Total Network Traffic as a Function of Simulation Time

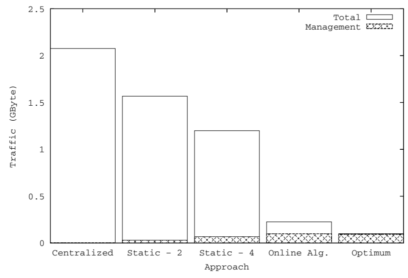

Fig. 8. Portion of Traffic due to Management Operations

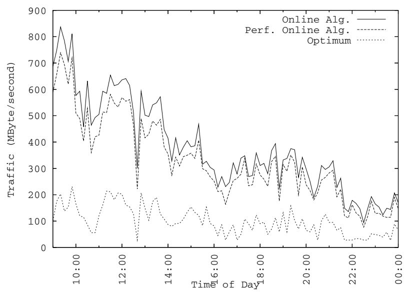

Fig. 9. Precision of Forecasting Method

rameters equal to 10 KBytes and 2000 mobile users. For the static approaches, the management traffic, due to replica maintenance operations, is a tiny fraction of the total traffic. The dynamic approaches reduce total traffic significantly by performing replication and thus, have a higher management traffic overhead. In particular, practically all the traffic generated in the offline optimal solution is due to management operations. Thus, the replication decisions made are payed off by significant reductions in indirect request service. The online dynamic algorithm generates approximately the same management overhead, with a total traffic that is only 126% higher than the offline optimal.

Forecasting Method

Finally, we evaluate the accuracy of the Double Exponential Smoothing forecasting method by comparing it with perfect forecasting, where the real client demands for the future δ periods are known *a priori*. We compare the performance of the online algorithm using either method, for a configuration with 12000 users and replication, indirect service and maintenance sizes equal to 2MBytes, 20KBytes and 900KBytes, respectively. We run the simulator for 150 periods, covering almost a whole day, and use δ equal to 7, for both methods. Figure 9 shows the total network traffic obtained as a function of the time of the simulated day. For comparison purposes, it also shows the results obtained with the offline optimal solution. Perfect forecasting reduces total traffic in only up to 9.7%, compared to the Double Exponential Smoothing method. Thus, this demand forecasting method is highly accurate. Note that, as time goes by, the online algorithm approximates to the offline optimal, being within 122% of optimal after 150 periods. Similar results were obtained with other values of δ.

5 Conclusions and Future Work

This papers proposed WDCDNM, a Wireless Dynamic Content Distribution Network Model that takes both temporal and spatial variations in client demand into account to dynamically replicate content with the goal of minimizing total traffic over the backbone of wireless networks. Our WDCDNM decides to replicate a certain content or remove an existing replica based on the solution of a tradeoff between minimizing the total traffic generated by replication and replica maintenance operations and minimizing the traffic resulting from indirect request service.

We proposed and evaluated two implementations of WDCDNM: an offline optimal solution and an online heuristic, based on demand forecasting. We used a previously developed mobility simulator to evaluate the performance of our solutions, contrasting it with the performance of previous static centralized and distributed approaches. Compared to the best static approach, with 4 replica servers, our online algorithm reduces total network traffic in up to 78%, producing solutions that are within four times the ideal offline optimal.

Directions for future work include introducing capacity constraints in WDCDNM, performing a quantitative comparison of our online solution and the ACDN algorithm [14] and further optimizing the online dynamic placement algorithm.

References

1. S. Albers. Competitive Online Algorithms. *Mathematical Programming*, 5:3–26, 2003.
2. B. L. Bowerman and R. T. O'Connell. *Forecasting and Time Series an Applied Approach.* Wadsworth Company, 1993.
3. Y. Chen, R. Katz, and J. Kubiatowicz. Dynamic Replica Placement for Scalable Content Delivery. In *Proc. International Workshop on Peer-to-Peer Systems*, Cambridge, MA, March 2002.
4. M.R. Garey and D. S. Jonhson. *Computer and Intractability: A Guide to the Theory of NP-Completeness.* W H Freeman, 1979.
5. Y. Guo, Z. Ge, B. Urgaonkar, P. Shenoy, and D. Towsley. Dynamic Cache Reconfiguration Strategies for a Cluster-Based Streaming Proxy. In *Proc. Web Caching and Content Distribution Workshop*, Hawthorne, NY, September 2003.
6. S. Jamin, C. Jin, A. R. Kurc, D. Raz, and Y. Shavitt. Constrained Mirror Placement on the Internet. In *Proc. IEEE INFOCOM*, Anchorage, AK, April 2001.
7. J. Kangasharju, J. Roberts, and K. Ross. Object Replication Strategies in Content Distribution Networks. In *Proc. Web Caching and Content Distribution Workshop*, Boston, MA, June 2001.

8. M. Karlsson and C. Karamanolis. Choosing Replica Placement Heuristics for Wide-Area Systems. In *Proc. International Conference on Distributed Computing Systems (ICDCS)*, Hachioji, Tokyo, Japan, March 2004.
9. P. Krishnan, D. Raz, and Y. Shavitt. The Cache Location Problem. *IEEE/ACM Transactions on Networking*, 8(5):568–582, October 2000.
10. B. Li, M. J. Golin, G. F. Italiano, X. Deng, and K. Sohraby. On the Optimal Placement of Web Proxies in the Internet. In *Proc. IEEE INFOCOM*, New York, NY, March 1999.
11. G. R. Mateus, O. Goussevskaia, and A. A. F. Loureiro. Simulating Demand-Driven Server and Service Location in Third Generation Mobile Systems. In *Proc. Europar 2003*, Klagenfurt, Austria, August 2003.
12. P. Mirchandani and R. Francis. *Discrete Location Theory.* John Wiley and Sons, 1990.
13. L. Qiu, V. N. Padmanabhan, and G. M. Voelker. On the Placement of Web Server Replicas. In *Proc. IEEE INFOCOM*, Anchorage, AK, April 2001.
14. M. Rabinovich, Z. Xiao, and A. Aggarwal. Computing on the Edge: A Platform for Replicating Internet. In *Proc. Web Caching and Content Distribution Workshop*, Hawthorne, NY, September 2003.
15. P. Radoslavov, R. Govindan, and D. Estrin. Topology-Informed Internet Replica Placement. In *Proc. Web Caching and Content Distribution Workshop*, Boston, MA, June 2001.
16. M. Tariq, R. Jain, and T. Kawahara. Mobility-Aware Server Selection for Mobile Streaming Multimedia Content Distribution Networks. In *Proc. Web Caching and Content Distribution Workshop*, Hawthorne, NY, September 2003.
17. T. Yoshimura, Y. Yonemoto, T. Ohya, M. Etoh, and S. Wee. Mobile Streaming Media CDN Enabled by SMIL. In *Proc. WWW Conf.*, Honolulu, HI, May 2002.
18. H. Yu and A. Vahdat. Minimal Replication Cost for Availability. In *Proc. 21st ACM Symposium on Principles of Distributed Computing (PODC)*, Monterey, CA, June 2002.
19. G. K. Zipf. *Human Behavior and the Principal of Least-Effort.* Addison-Wesley, Cambridge, MA, 1949.