Wireless Network Compression: Code Design and Trade offs

Mohsen Sardari, Ahmad Beirami, Faramarz Fekri
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332
Email:{mohsen.sardari, beirami, fekri}@ece.gatech.edu

***Abstract*—Traces derived from the real-world mobile network traffic show significant inter-client redundancy among packets. This has inspired new solutions to reduce the amount of redundancy present in the data in order to manage the explosive traffic. In this paper, we propose a novel approach to leverage this redundancy by employing a novel network compression technique using wireless helper nodes. A Helper node provides the side information, that it obtained via overhearing, to mobile clients and assists the wireless gateway to more effectively compress each new packet when serving every new client. We investigate the potential benefits of wireless network compression from an information-theoretic point of view. We also describe a coding mechanism which adapts the two-part coding scheme to wireless network compression. We optimize this coding scheme to achieve minimum cost communication in the network. We also characterize the trade-off between the number of bits sent by the wireless gateway and the number of bits sent by the helper to a client.**

***Index Terms*—Memory-Assisted Compression, Redundancy Elimination, Wireless Networks, Overhearing.**

I. Introduction

Mobile data efficiency is an important feature of wireless communication. It increasingly draws attention as providers face the difficulty of handling the explosive increase in the demand and look for solutions to reduce the cost of data delivery in wireless networks. One potential solution is to find ways to eliminate the redundant data that is being transmitted to clients through the bottle-neck of the network, the most important being the last hop: the wireless link from the wireless gateway to the mobile client. As suggested in the following, there are two main dimensions that contribute to the redundancy within a network, first redundancy in the content and second redundancy across different clients. IP-layer redundancy elimination, in the form of repetition suppression for a single client, has been able to save up to 60% bandwidth on the last hop links [1]. These saving are obtained in the first dimension of redundancy. Further, recent studies show in [2] that traces derived from real-world wireless traffic collected in a noise-free environment contain around 50% inter-client repetition within packets, i.e., duplicate strings across packets. All these signify the importance of redundancy elimination in the flows. However, all the existing works [1], [2] confine themselves to deduplication of repeated patterns for redundancy elimination. It is expected to reduce the redundancies even more significantly by information-theoretic compression methods such as dictionary-based and statistical approaches. The proposed wireless network compression in this paper is focused on these information-theoretic methods that explore network memory.

In our previous work [3]–[5], we have taken the first steps towards characterizing the achievable benefits of exploiting the packet redundancies beyond simple repetition suppression. Data compression and source coding are natural candidates for this task. In [3], [4], we have formulated the redundancy elimination as *network compression via network memory* and introduced a new framework for compression of network data called *memory-assisted compression*. This approach significantly departs from the traditional source coding techniques in that it relies on the network memory for compression. We have already explored the network-wide gain of memory-assisted compression in wired networks. However, the gain of memorization and memory-assisted compression is more spelled out in the bandwidth-constrained wireless networks. On the other hand, it is more challenging to establish the memorization scheme in such networks and our solution for wired networks is not applicable to wireless networks. This is mainly because the objective of the compression scheme in wireless environment should be to off-load the wireless gateway so as to enable it to serve more clients while guaranteeing that all packets are recoverable in a strictly lossless manner at the clients. Further, the effect of error-prone wireless links should be studied on memory-assisted compression.

In this paper, we study wireless network compression. In particular, we explore the benefits of memory-assisted compression in the last hop wireless links from the wireless gateway to the mobile clients by deploying memory-enabled helpers. We focus on total throughput enhancement and gateway off-loading. This would be of particular interest for WiFi and cellular networks. We stress that our scheme is universal in the sense that we do not assume to know the distribution of the source traffic a priori.

We propose to position data redundancy elimination via network compression by deploying some nodes as helpers to overhear previously transmitted packets from the wireless gateway to mobile clients. These overhearing packets provide statistical information about the traffic. Then, in the compression of a new packet, this information is sent from overhearing (helper) node to a mobile client to supplement (as a side

This material is based upon work supported partially by the National Science Foundation under Grant No. CNS-1017234.

information) the compressed data from the wireless gateway to the mobile client; enabling the client to decompress the codeword and recover the packet. Since the communication in the link between the overhearing memory-enabled helper and the client is by far less costly than that of the wireless gateway and the client, the network compression via overhearing nodes is proposed, by design, to reduce traffic on the link from the wireless gateway to the mobile client. In this paper, we aim to study both analytically and experimentally the fundamental limits of the wireless network compression via overhearing (memory-enabled) nodes.

To motivate the need for memory-enabled helpers, we first demonstrate using an experiment that conventional compression techniques (which do not use the side information obtained from memory) perform poorly on the Internet traffic data. We gathered some packets from CNN web server and chose two different types of universal compression algorithms for the experiment: 1) The statistical compression method (e.g., Context Tree Weighting (CTW) [6]), and 2) The dictionary-based compression method (e.g., LZ algorithm [7]). As shown in Fig. 1, a modest compression performance can be achieved by compression of a packet when the packet length n is small to moderate size. For example, for a data packet of length n = 1kB, the compression rate is about 5 bits per byte. Note that the uncompressed packet requires 8 bits per byte for representation. We also note that as the packet length n increases, the compression performance improves. For very long packets, the compression rate is about 0.5 bits per byte. In other words, comparing the compression performance between n = 1kB and n = 16MB, there is a penalty of factor 10 on the compression performance (i.e., 5 as opposed to 0.5). Since an IP packet is approximately 1500 bytes in practice, there is a huge penalty paid by the naive compression of a packet. The compression performance loss attributed to the finiteness of packet length can be removed using the memory-assisted compression framework described in [5]. In other words, memory can be used to compensate for the finiteness of the packet length and improve the compression as quantified below as the memorization gain. We denote a packet of length n as $x^n = (x_1, \ldots, x_n)$ where each x_i is a byte. Let $\mathbf{E}l_n(X^n)$ be the expected compressed length of packet x^n when compressed by itself and let $\mathbf{E}l_{n|m}(X^n)$ be the expected code length for x^n when compressed using a memorized sequence of length m as a side information about the packet x^n.[1] The gain of memory-assisted compression $g(n, m)$ is defined as

$$g(n, m) \triangleq \frac{\mathbf{E}l_n(X^n)}{\mathbf{E}l_{n|m}(X^n)}.$$

The gain g for the example above is depicted in Fig. 2 for both schemes. We see an average gain of 2.5 can be expected on compressing an IP packets with a modest 4MB memory length using memory-assisted CTW algorithm. Note that this memorization gain is measured with respect to the conventional compression schemes.

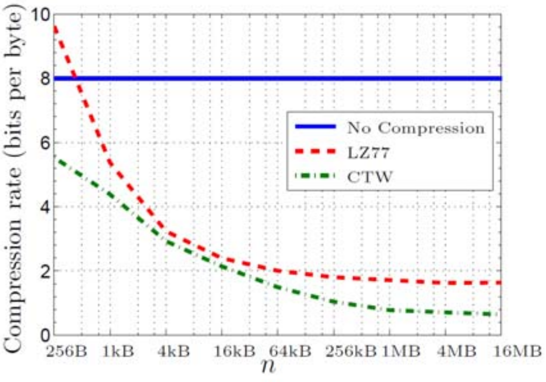

Fig. 1. The compression rate of a sample web trace (obtained from CNN web server) as a function of the sequence length, obtained using LZ77 and CTW compression algorithms.

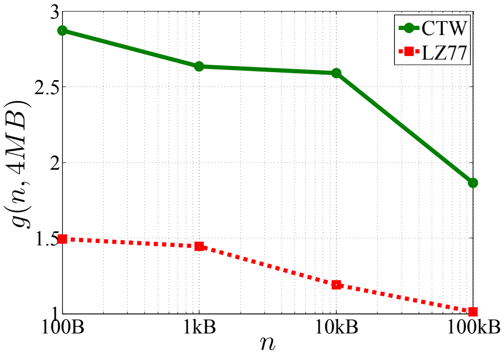

Fig. 2. The gain g of memory-assisted compression, for memory size of 4MB, for CTW and LZ77 compression algorithms.

The rest of the paper is organized as follows. In Sec. II, we present application of memory-assisted compression for a sample wireless network scenario. We describe an abstraction of the compression problem and present a practical code design, and analyze the performance of the code in Sec. III. Simulation results, performed in NS-2 simulator, are provided in Sec. IV. Discussion on the complexity of the proposed code design is given in Sec. V. Finally, Sec. VI concludes the paper.

A. Related Work

The redundancy elimination techniques are mostly based on application-layer caching mechanisms. However, application-layer caching is designed to remove redundancy based on the popularity of the entire contents. Therefore, it cannot be effective as most of the traffic redundancy is present at the packet and sub-packet levels [8]–[11]. Hence, a new line of research advocating network layer redundancy elimination (RE) has attracted a lot of attention. Recently developed RE algorithms reduce traffic volume on bandwidth-constrained network paths by avoiding the transmission of repeated byte sequences in the network layer. In other words, whenever a segment of the packet's content already exists at the mobile node, the

[1] In this paper, X^n denotes a random sequence of length n and x^n denotes a packet which is the realization of X^n.

RE algorithms residing in the backbone (or the access-point) would replace that segment with a pointer, and hence, reducing the data transmission. As opposed to the *network compression* technique used in this work, RE algorithms [1], [2] are focused on eliminating the repeated data segments and do not fully leverage the statistical properties of the data stream.

The proposed network compression principle is inspired by the content-centric design [12] and attempts to reduce redundancy using the content of the previous packets. However, a naive application of off-the-shelf universal compression algorithms on the packets (i.e., end-to-end compression) does not alleviate the problem for two main reasons: 1) simple end-to-end approaches overlook the redundancy present *across* different clients in the networks, hence not very effective for wireless schemes, and 2) end-to-end universal compression of packets results in poor compression performance due to the finiteness of the packet length, as explained in the previous section. The network compression (via helpers) proposed in this work overcomes these shortcomings of traditional universal compression.

II. Redundancy Elimination in Wireless Networks via Memory-Assisted Compression

In this work, we will focus on non-mobile overhearing memory-enabled helpers. As shown in Fig. 3, for an example scenario involving a single wireless gateway S, a mobile client C and a helper M, the idea is to deploy memory-enabled helpers that are capable of overhearing communication from the wireless gateway to all the mobile clients inside the coverage area of the wireless gateway. The overhearing comes at no extra cost due to the broadcast nature of the wireless communication. Although this can be applied to every cellular or WiFi access networks, one realization of such memory-enabled helpers can be in femto-cell network designs combined with traditional macro-cell networks, as in [13]. The proposed network compression via memory at the overhearing nodes works as following. First, recall that the traffic (i.e., the packets) destined to different mobile clients from the gateway S are highly correlated. Therefore, the overhearing memory-enabled helpers can overhear the past communication between the cell tower (or the WiFi access-point) and mobile nodes and hence, learn the statistical properties of the packets in the traffic. These extracted statistical properties can then be used as a side information (if provided to the client) improving the compression performance on the future traffic from the gateway S to any mobile client. In other words, the memory-enabled helpers can possibly help to reduce the transmission load of the cell tower by transmitting the side-information about the data traffic to the clients using a *less costly* memory-client M-C link.

Since the objective of network compression is to minimize the load of wireless gateway to support more clients, we can define a virtual cost for S-C and M-C links in Fig. 3. Let κ denote the ratio of the cost of communicating one bit in the S-C link to that of the M-C link. In practical settings, it is rational to assume that the S-C link is much more costly than the M-C link. Hence, κ is much greater than unity. We use this κ in our analytical development to minimize the aggregate cost of communication. This departs from the objective of only minimizing the total number of bits transmitted from a source to a destination in a traditional setup. It is imperative to note that the memory-assisted compression via overhearing memory-enabled helpers would provide additional compression benefits over and beyond those already gained by simple end-to-end compression, i.e., compressing the packet from S to C while there is no memory deployed. Further, the proposed network compression only entails negligible extra computational overhead at the wireless gateway and the overhearing memory-enabled helper while reducing the aggregate cost.

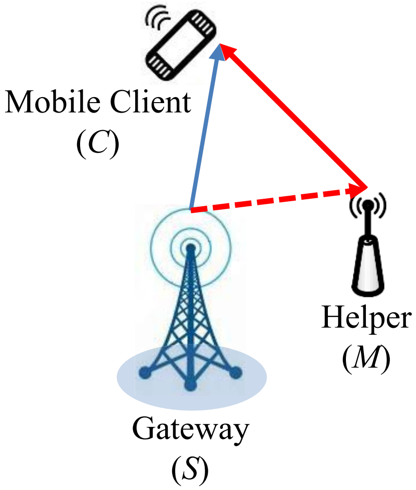

Fig. 3. An illustrative example of a wireless network with a single helper (deployed memory-enabled helper). A short-lived connection to the source by a mobile client is shown by a solid arrow. Overhearing is shown by a dashed arrow. The link supplementing side-information is shown by a thick solid arrow.

A. Setup

The abstract model of the network compression via overhearing memory-enabled helper is shown in Fig. 4, for a single helper and a client. We consider the traffic reduction (compression) over the down-link. The data are delivered from the wireless gateway S, which is the source in our abstraction, to the mobile client C. We only consider schemes where the sequence x^n, i.e. the packet, is exactly recoverable at the destination. Therefore, all the compression schemes considered are strictly lossless.

Definition 1 *Let $\mathcal{A}^n$ be the set of all sequences of length n over alphabet $\mathcal{A}$. The code $c_n(\cdot) : \mathcal{A}^n \to \{0,1\}^*$ is called strictly lossless if there exists a reverse mapping $d_n(\cdot) : \{0,1\}^* \to \mathcal{A}^n$ such that*

$$\forall x^n \in \mathcal{A}^n : \quad d_n(c_n(x^n)) = x^n.$$

All of the practical data compression schemes are examples of strictly lossless codes, namely, the arithmetic coding, Huffman coding, LZ algorithm, and the Context-Tree-Weighting (CTW).

The source is assumed to generate and send different packets to mobile nodes one at a time (unicast). The gateway covers the entire area, hence, the overhearing memory-enabled helpers are also capable of overhearing the communication from S to client C. Each overhearing memory-enabled helper is also assumed to be capable of sending information to those mobile nodes in its vicinity.

The link between S and C is lossy due to the wireless channel but we assume a proper feedback for packet retransmission would take care of packet losses on the S-C link. Note that in practice, we consider a stable situation which is when the transitional memorization phase is over. In other words, we assume that every overhearing node has been in the network for a long time and has accumulated sufficient knowledge about the source model from all the past communication (the memorization phase). To proceed with our formulation of the problem, let assume that the concatenation of all delivered packets from S to several mobile clients, during transition phase, is a memorized sequence of length m denoted as y^m. In practice, it is rational to assume that the memory-enabled helper has observed a sufficient number of packets (when S was serving several other clients), and hence, the total size m of memorized packets is assumed to be sufficiently large. This assumption is not necessary for network compression but it would simplify our presentation.

An Example Scenario: The basic principle in network compression in Fig. 4 can be described as following. For the moment, assume that both M and S share exactly the same memory y^m after the memorization phase. Further, assume we use a dictionary-based compression method such as LZ77 as in DEFLATE [7]. The LZ77 algorithm would form a dictionary of codewords that are able to describe the packets from S using the memory y^m. Then, in the compression of a new packet x^n, the server S would only send (to the client C) the address of the codeword in the dictionary which would be complemented by the memory-enabled helper M who would forward (to the mobile client C) the corresponding codeword upon overhearing the address. Hence, the mobile client would be able to decode and recover x^n although the client did not have memory (i.e., the dictionary). In this scenario, the cost of sending a short address (on the link S-C) would be very low relative to the cost of sending the long codeword on the link from the memory-enabled helper to the client. This would achieve the principle objective of network compression which is saving the cost on the link from the wireless gateway S to the mobile client. In the above example, we simplify the description of LZ77. In reality, we clarify that several addresses and codewords are transferred during delivery of x^n, however, the main benefit of network compression remains intact. The above example via LZ77 can be generalized to the other compression schemes as we will discuss later in the paper.

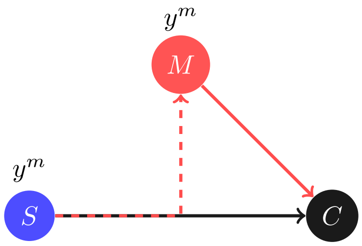

Fig. 4. The abstract illustration of the traffic reduction problem via network compression. The memorized sequence y^m represents the total past data overheard by M from S to the clients.

As mentioned before, since we wish to reduce the load of the gateway, we have an asymmetric situation where we assign a higher cost to the channel from the source to the client than from the helper to the client. This asymmetry between the channel costs is motivated by real-world cellular networks where a single base-station serves a large number of clients. Hence, if the load of the base-station by each client is reduced, it can potentially serve a larger number of clients. For example, the S-C link from the base-station to the client (and hence the overhearing link S-M) can operate in a frequency different from the M-C link. Whenever the base-station hands-off the connection to the mobile client (and the overhearing memory-enabled helper), its frequency slot frees up and a new client can be served. Further, due to lower communication radius, the frequency slot allocated to the M-C link can be reused within a cell for the link between some other memory-enabled helper with another client. This architecture together with the proposed network compression offers a significant opportunity for traffic reduction so as to deliver x^n by exploiting the side-information y^m shared between S and M.

Let x^n be a packet of length n that is delivered from the source to C. To proceed the network compression problem in wireless networks, we present the problem as in Fig. 4. That is, the memorized sequence y^m is available at both S and M. The problem of interest is as to how the encoder of S would encode x^n such that the aggregate communication cost on the link S-C together with the cost of supplemented bits on the link M-C would be minimized, provided that x^n would be recovered at the client.

III. Code Design for Network Compression via Overhearing Helper

Due to superiority of statistical codes with respect to the dictionary-based compression schemes (e.g., LZ77), in the rest of this paper, we focus on the former. The main feature of our approach is that the source can rely on the memory-enabled helper to send side-information to clients. In the statistical universal compression technique, this side information is in fact the source model formed at the overhearing memory-enabled helper using the sequence y^m. Now, the question is as to how this side information can be used for improving the efficiency of compression at the source. We propose a two–part coding scheme which is an adaptation of two–part universal codes in the source coding literature (c.f. [14], [15] and the references therein) to the network compression via overhearing memory-enabled helpers. Next, we study the efficiency of the two–part code, introduce the necessary notations and describe its adaptation.

A. Two–Part Code

For the analysis, we assume that S is a parametric source. Let μ_θ be the probability density function of the source depending on a d-dimensional parametric vector θ which takes values in $\Theta \subset \Re^d$. Consider a parametric source with probability density function μ_θ. By this setup, for example, for a binary Bernoulli (memoryless) source with parameter γ, the probability that the source would output x^n, with k ones and $n-k$ zeroes, is given by $\mu_\gamma(x^n) = \gamma^k(1-\gamma)^{n-k}$. If the parameter vector $\theta \in \Theta$ was known, the ideal code length of a packet x^n, obtained from the Shannon code, would be $\log 1/\mu_\theta(x^n)$. Since in practice we do not have any prior knowledge of θ, we have to encode the packet with a universal distribution $P(x^n)$. Hence, we have to use more number of bits to encode the packet. This overhead is called *code redundancy* and is defined as

$$R(P, \mu_\theta) = \mathbf{E}[l(X^n)] - H_n(\theta) = \mathbf{E}\left[\log \frac{\mu_\theta(X^n)}{P(X^n)}\right], \quad (1)$$

where $l(x^n)$ is the length of the codeword assigned to x^n by the universal encoder. Further,

$$H_n(\theta) = \mathbf{E}\left[\log \frac{1}{\mu_\theta(X^n)}\right]$$

denotes the source entropy which is the fundamental limit of compression for the source.

Code redundancy stands at the root of all methods of measuring the performance of universal codes; redundancy compares the code length assigned by universal procedures to the best code length $\log 1/\mu_\theta(x^n)$. It is important to note that for short length packets, there is a large gap between the code length of the best universal code and the source entropy, i.e., the average code redundancy is considerable. We are after practical designs for $P(\cdot)$ which has close to optimal universal code lengths and is easy to implement in our wireless network problem. One good candidate is the two–part code. It can be shown that the redundancy of the two–part codes approaches the main term of $\frac{d}{2}\log n$ with a negligible $O(1)$ overhead [15]–[17].

The two–part code is a source coding scheme composed of two parts that can be crudely described as follows: the first part tries to obtain the best estimation, i.e. $\hat{\theta} \in \Theta$, along with a code for the parameter vector θ of the source from an observed packet. This estimate $\hat{\theta}$ is then used in the second part for compression of the packet. The closer the estimate $\hat{\theta}$ gets to the Maximum Likelihood (ML) estimate θ_{ML} of the source parameter θ, the smaller the description length of the packet to be compressed becomes (i.e., better compression). However, a better estimate of θ would need more bits for the description of $\hat{\theta}$. To achieve the best code length, one should use an estimate of θ that minimizes the total code length of the two–part code. Hence, the two–part code length is given by

$$l(x^n) = \min_{\hat{\theta}\in\Theta}\{l(\hat{\theta}) - \log P_{\hat{\theta}}(x^n)\}, \quad (2)$$

where $l(\hat{\theta})$ is the universal length of the codeword describing the estimate $\hat{\theta}$ and $-\log P_{\hat{\theta}}(x^n)$ is the description length of a packet x^n given the estimate $\hat{\theta}$.

TABLE I
SUMMARY OF WIRELESS NETWORK COMPRESSION VIA TWO–PART CODES.

Initialization (S and M)
The helper node M overhears the communication of S with past clients and accumulates knowledge about the source model and its statistics. Node S also performs the same operations to construct the model. The total sequence size observed by M is y^m.
Operation (S)
For every new packet (sequence) x^n, S uses the statistical model to estimate the probability of the symbols in x^n. Then, these probabilities estimates along with x^n are sent to an encoder (e.g., an arithmetic encoder). The output of the encoder (NOT the probability estimates) is then sent to C.
Operation (M)
Once the helper M finds out that the compressed packet $c(x^n)$ is sent to a client within its coverage, then M sends the probability estimates necessary for decompression to C.
Operation (C)
The client C receives the output of the (arithmetic) encoder from S and the probability estimates from M and feeds them to a decoder (e.g., arithmetic decoder) to reconstruct x^n.

Using previous results in the literature, we can obtain the best code length and redundancy for the simple two–part codes. Our objective is to use those results in the context of the proposed memory-assisted compression to off-load the source via overhearing memory-enabled helpers. As such, to proceed we need to consider the asymmetric cost of transmission in our setup. From (2), $\mathbf{E}[l(X^n)]$ is the expected length of sequence sent by source when the source compresses the packet x^n without regard to the memory-enabled helper. On the other hand, by exploiting the memory-enabled helper, the source can send a codeword of size close to $H_n(\theta)$, the entropy of the packet. Both the client and the memory-enabled helper receive this codeword. However, the client cannot decode the codeword as it does not know the source parameter. On the other hand, the memory-enabled helper M knows the source parameter. To guarantee decodability, the memory-enabled helper then sends the codeword corresponding to the estimate of the source parameter to the client, extracted from y^m, with length $l(\hat{\theta}|y^m)$. Therefore, the total cost of transmission would be close to $\kappa H_n + l(\hat{\theta}|y^m)$.

More precisely, the cost of delivering x^n given the memorized sequence y^m with length $m \gg n$ at the memory-enabled helper, is given by

$$\mathcal{C} = \min_{\hat{\theta}\in\Theta}\{l(\hat{\theta}|y^m) - \kappa \log P_{\hat{\theta}}(x^n|y^m)\}. \quad (3)$$

The expected cost, i.e., $\mathbf{E}[\mathcal{C}]$, would then determine the effective cost of communication. In the next section, we investigate the trade-offs of the code design and characterize the achievable communication cost.

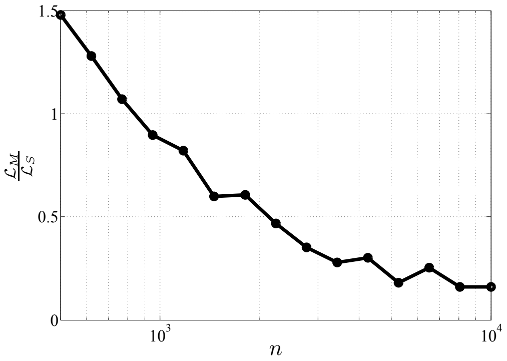

Fig. 5. Ratio of the size of the output of the helper $\mathcal{L}_M$ to the size of the source output $\mathcal{L}_S$ vs. the size of packet n for a memoryless source with alphabet size 256.

B. Performance Evaluation of Two–Part Memory-Assisted Compression

In order to characterize the performance of two–part coding, we consider a two–part coding scheme that attributes b bits to identify an estimate for the unknown source parameters, i.e., we set $l(\hat{\theta}|y^m)$ to be b bits. Therefore, there would be 2^b possible estimate points in the truncated parameter space $\Phi = \{\theta_1, \ldots, \theta_{2^b}\}$ for the identification of the source parameter. We obtain the following theorem regarding the communication cost when we use a two–stage coding scheme. Let $\Gamma(\cdot)$ be the Euler's gamma function, and $|I(\cdot)|$ be the determinant of the Fisher information matrix [14].

Theorem 1 *Let $\phi \in \Phi$ be the estimate of the d-dimensional source parameter vector θ. Then, with probability equal to one, the total cost of communication is obtained as*

$$\mathcal{C} = b + \kappa \left[H_n(\theta) + \frac{n\omega}{2^{1+2b/d}} \right]. \qquad (4)$$

where the constant ω is equal to

$$\omega = \left(\frac{\int |I(\theta)|^{\frac{1}{2}} d\theta}{C_d} \right)^{2/d} \log e.$$

Here, C_d is given by $C_d = \frac{\Gamma(\frac{d}{2})^d}{\Gamma(\frac{d}{2}+1)}$.

Proof: Proof is provided in the extended version [18]. ■

To illustrate the trade-offs in Thm. 1, we make the following observations by studying different terms in $\mathcal{C}$. As mentioned previously, the most important parameter of the design is the number of bits b allocated for estimation of the parameter. Using b bits, let $\hat{\phi} \in \Phi$ be the estimate of the source parameter that minimizes the code length. If the cost of communication is the same for all the links, that is $\kappa = 1$, we have

$$\begin{aligned} \mathbf{E}[\mathcal{C}]_{|\kappa=1} &= H_n(\theta) + R(P_{\hat{\phi}}, \mu_\theta) + b \\ &= H_n(\theta) + \mathbf{E}\left[\log \frac{\mu_\theta(X^n)}{P_{\hat{\phi}}(X^n)} \right] + b. \qquad (5) \end{aligned}$$

TABLE II
SIMULATION PARAMETERS AND VALUES

Parameter	**Value**
Number of helpers (M)	$0 - 10$
Comm. Radius of S	250m
Comm. Radius of helper	20m
Memory-Assisted gain g for one client	1.5–4
CBR over UDP rate	64 kbps
UDP baseline packet size	8000 bits
Packet Drop Rate Threshold	10%

From (5), we see that $\mathbf{E}\left[\log \frac{\mu_\theta(X^n)}{P_{\hat{\phi}}(X^n)}\right] + b$ is the code redundancy of two–stage compression scheme. As b increases, the code redundancy of the second stage of the two–stage code, i.e. $\mathbf{E}\left[\log \frac{\mu_\theta(X^n)}{P_{\hat{\phi}}(X^n)}\right]$, decreases. The interplay between these two terms determines the total number of transmitted bits from the source and the memory-enabled helper. Let $\mathcal{L}_S$ be the total number of bits sent by S and $\mathcal{L}_M$ be the total number of bits sent by M to the client. From (5), we have

$$\begin{cases} \mathcal{L}_S &= H_n(\theta) + \mathbf{E}\left[\log \frac{\mu_\theta(X^n)}{P_{\hat{\phi}}(X^n)}\right] \\ \mathcal{L}_M &= b \end{cases}.$$

Fig. 5 shows the ratio $\frac{\mathcal{L}_M}{\mathcal{L}_S}$ for a memoryless source model with alphabet size 256. Note that since the packet length is short, a memoryless source model is adequate for modeling of the underlying source in practice. The graph in Fig. 5 is generated using a uniform discretization of the parameter space. Further, the packets are compressed using a standard arithmetic coder using the estimate of the parameter as a side information.

We have used the result of Fig. 5 later in the simulation section to determine the output rate of source and helpers to clients. For example, for a packet length of 1kB, the size of the parameter estimate is roughly the same size as the compressed packet, i.e., $\frac{\mathcal{L}_M}{\mathcal{L}_S} \approx 1$.

IV. SIMULATION

A. Simulaton Setup

To evaluate the performance of the proposed memory-assisted compression via helpers, we used NS-2 simulator [19]. We employed a flat grid topography with a wireless base-station (S) at the origin. Further, multiple memory-enabled helpers (M) are deployed within the coverage of S. The helpers are uniformly distributed in the coverage of S, which is assumed to be a circle of radius 250m. The communication range of the helpers is 20m and they are placed such that they are outside of the communication range of each other. All the mobile clients are within the communication range of S, but only some of them are covered by helpers at any time.

We simulate constant bit rate (CBR) traffic generator over user datagram protocol (UDP). We have considered the case where S shares a common memory with each of the helpers and that memory is used for compression of packets sent to mobile nodes within the coverage of the corresponding helper. Further, each mobile client (if covered by the helper) only

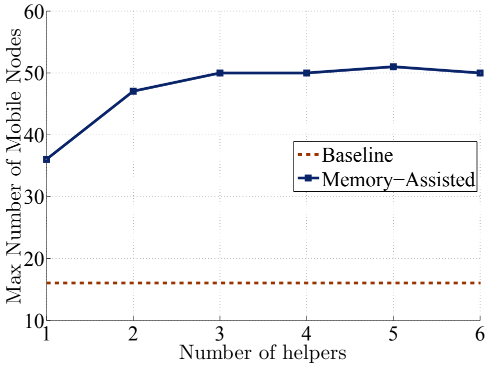

Fig. 6. Maximum number of mobile nodes supported by S vs. the number of helpers in the network. The packet drop rate threshold is fixed at 10% and the traffic generator is CBR over UDP, as in Table II.

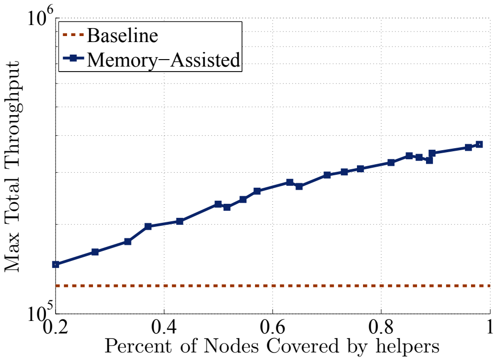

Fig. 7. Maximum total throughput in the network vs. the fraction of mobile nodes covered by helpers for UDP.

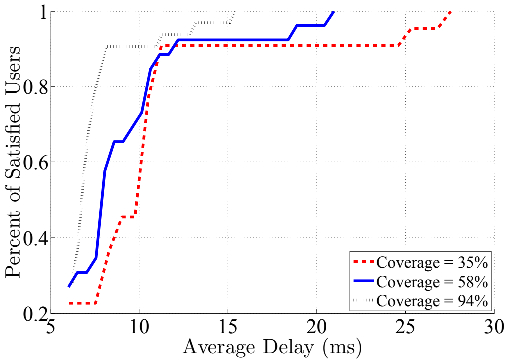

Fig. 8. Fraction of satisfied users in the network vs. maximum allowed average delay of packets for UDP traffic for different helper coverage percentages.

helped by a unique overhearing node. Obviously, the memory is not used for compression of packets sent to nodes that are not in the range of any helper and receive the packet directly from S. For the baseline simulation scenario, we consider the case where no helper is deployed and all the communication is conducted by S and packets are compressed individually (end-to-end compression).

Unless otherwise specified, the memory-assisted gain is chosen to be $g = 2$. As evident from Fig. 2, this is expected for packets of sizes around 1kB. The details of simulation parameters are given in Table II.

B. Simulation Results

To examine the effectiveness of the memory-assisted compression, with respect to baseline scheme, we have considered three quantities and evaluated them for UDP scenario. The first quantity is the maximum number of nodes that can be supported, for the traffic described in Sec. IV-A, in a network given a packet drop rate threshold, which is the maximum acceptable drop rate. We observe that using memory-assisted compression the maximum number of nodes increases from 15 to almost 50, as shown in Fig. 6. Since the bottleneck of the network is the output bandwidth of S, adding helpers beyond a certain number does not increase the maximum number of nodes supported.

In Fig. 7, we have depicted the maximum total throughput versus the fraction of the nodes covered by helpers. As expected, as helpers cover more mobile nodes in the network, higher total throughput is achieved.

The third quantity of interest is the Quality of Service (QoS). To demonstrate the benefit of memory-assisted compression on QoS, we have considered a simulation scenario with fixed number of clients and measured the average delay of packets for each client. Fig. 8 depicts the fraction of satisfied clients for a given maximum allowable average delay. As we see, users experience less amount of delay as the fraction of nodes covered by helpers increase.

V. Discussion

In this section we want to briefly discuss the complexity of the proposed two–part coding strategy described here. As described earlier, the first stage (i.e., forming the model) involves finding the best description of the information source using the memorized sequence of length m. This stage has a complexity linear in size of m. The second stage which is the actual compression of a new packet involves entropy coding of a packet of size n which has linear complexity in the packet size (i.e., n). With regard to the cost of communication, in this paper we assumed that the cost of transmitting one bit in the M-C channel is unity and S-C channel is κ times more costly. In a practical setting these costs can be assigned through examination of power and bandwidth constraints and our framework can be employed accordingly.

VI. Conclusion

In this paper, we introduced wireless network compression, a new method for decreasing the output flow of the wireless gateway in a wireless network by eliminating redundancy from the traffic. The key idea is to deploy a number of memory-enabled helpers in the coverage area of the wireless gateway that are capable of overhearing and memorizing previous communications on the down-link from the wireless gateway to mobile nodes. These helpers provide side-information to mobile clients that enables the wireless gateway to send fewer bits to the client by a proposed memory-assisted compression technique just above layer 3. We adapted the proposed memory-assisted compression with the asymmetric cost of communication from the wireless gateway to the client (S-C) versus the memory-enabled helper to the client (M-C) and arrived at optimal two–part codes for the compression. The NS-2 simulation results show that network compression holds a great promise for improving the data transmission efficiency in wireless networks. We observe that network compression increases the maximum throughput significantly while reducing the average delay of packets (hence better QoS) for UDP traffic.

References

[1] C. Lumezanu, K. Guo, N. Spring, and B. Bhattacharjee, "The effect of packet loss on redundancy elimination in cellular wireless networks," in *Internet Measurement Conference*, 2010.

[2] S. Hsiang-Shen, A. Gember, A. Anand, and A. Akella., "Refactoring content overhearing to improve wireless performance," in *MobiCom*, Las Vegas, NV, 2011.

[3] M. Sardari, A. Beirami, and F. Fekri, "Memory-assisted universal compression of network flows," in *IEEE INFOCOM*, Orlando, FL, March 2012, pp. 91–99.

[4] ——, "On the network-wide gain of memory-assisted source coding," in *2011 IEEE Information Theory Workshop (ITW)*, October 2011, pp. 476–480.

[5] A. Beirami, M. Sardari, and F. Fekri, "Results on the fundamental gain of memory-assisted universal source coding," in *2012 IEEE International Symposium on Information Theory (ISIT '2012)*, July 2012, pp. 1092–1096.

[6] F. Willems, Y. Shtarkov, and T. Tjalkens, "The context-tree weighting method: basic properties," *IEEE Trans. Info. Theory*, vol. 41, no. 3, pp. 653–664, May 1995.

[7] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Info. Theory*, vol. 23, no. 3, pp. 337–343, May 1977.

[8] Z. Zhuang, C.-L. Tsao, and R. Sivakumar, "Curing the amnesia: Network memory for the internet," Tech Report, 2009. [Online]. Available: http://www.ece.gatech.edu/research/GNAN/archive/tr-nm.pdf

[9] N. T. Spring and D. Wetherall, "A protocol-independent technique for eliminating redundant network traffic," *SIGCOMM*, vol. 30, no. 4, pp. 87–95, 2000.

[10] A. Anand, V. Sekar, and A. Akella, "Smartre: an architecture for coordinated network-wide redundancy elimination," *SIGCOMM*, vol. 39, no. 4, pp. 87–98, 2009.

[11] A. Anand, A. Gupta, A. Akella, S. Seshan, and S. Shenker, "Packet caches on routers: the implications of universal redundant traffic elimination," *SIGCOMM*, vol. 38, pp. 219–230, 2008.

[12] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. Briggs, and R. Braynard, "Networking named content," in *Proceedings of the 5th ACM CoNEXT*, 2009, pp. 1–12.

[13] V. Chandrasekhar, J. Andrews, and A. Gatherer, "Femtocell networks: a survey," *IEEE Comm. Magazine*, vol. 46, no. 9, pp. 59–67, 2008.

[14] P. D. Grunwald, *The minimum description length principle*. The MIT Press, 2007.

[15] A. Beirami and F. Fekri, "Results on the redundancy of universal compression for finite-length sequences," in *IEEE Intl. Symp. Info. Theory (ISIT)*, Jul 31-Aug 5 2011, pp. 1504–1508.

[16] A. R. Barron and T. M. Cover, "Minimum complexity density estimation," *IEEE Trans. Info. Theory*, vol. 37, no. 4, pp. 1034–1054, 1991.

[17] J. Rissanen, "Strong optimality of the normalized ML models as universal codes and information in data," *IEEE Trans. Info. Theory*, vol. 47, no. 5, pp. 1712 –1717, July 2001.

[18] http://users.ece.gatech.edu/msardari3/ITA13extended.pdf.

[19] "The Network Simulator NS-2," http://www.isi.edu/nsnam/ns/.