TANGO: Toward a More Reliable Mobile Streaming through Cooperation between Cellular Network and Mobile Devices

Nawanol Theera-Ampornpunt* Tarun Mangla† Saurabh Bagchi*
Rajesh Panta‡, Kaustubh Joshi‡, Mostafa Ammar†, and Ellen Zegura†
*Purdue University, †Georgia Institute of Technology, ‡AT&T Labs Research

***Abstract*—Multimedia streaming is a major mobile application, accounting for more than half of total mobile traffic. Streaming applications usually have a static buffering strategy. For example, buffer size is limited to x minutes of the stream, where x is optimized to provide the best trade-off between minimizing stalls and limiting waste of user's bandwidth and energy resulting from user abandonment. We show that such strategies based on information available on the mobile device alone do not work well when network conditions change dynamically, *e.g.*, connectivity degrades due to congestion. We propose an alternative strategy using the framework called TANGO, based on a novel idea of cooperation between cellular network and mobile devices. By monitoring real-time network conditions and continuously predicting user location, our system is able to predict connectivity degradation in the near term. In such events, a notification is sent to the mobile device so that the streaming application can initiate a mitigation action, such as to pre-cache more content. In simulations based on real user traces, we found that TANGO reduces pause time by 13–72%, significantly outperforming DASH, which is the state of the art.**

I. INTRODUCTION

Multimedia streaming has become a major application for mobile users. Cisco reports that mobile video traffic accounted for 55% of total mobile traffic in 2014 [1]. In 2012, 70% of Pandora's usage was from mobile devices [6]. However, the growth of mobile network capacity is not keeping pace with the growth of traffic. Cisco estimates that mobile traffic will increase 10-fold between 2014 and 2019. On the other hand, the observed growth in network capacity has been much slower, at 4-fold per 5 year on average [2]. Therefore, network congestion will be more and more common in the future, and mobile applications will need better mechanisms to mitigate the problem.

Streaming applications usually limit the download rate by setting a maximum buffer size and/or limiting the download rate. A buffer that is too large would result in unnecessary energy and bandwidth usage due to unpredictable user abandonment. A buffer that is too small would result in interrupted playback in the event of short-term network connectivity problems. Existing systems such as adaptive bit-rate schemes use a "static" buffer and try to keep the buffer full by adapting the bit rate according to current network conditions. In this paper, we argue that the static buffer approach has fundamental limitations and is not adequate to address network connectivity problems. If the network condition is good (the common case), the buffer should be small. If the network condition is bad, the buffer should be large—directly proportional to the duration for which the network is going to be in the degraded state. However, the mobile device alone has no way of predicting future network quality. The cellular network, on the other hand, has a global view of the network. It knows which areas in the potential path of the streaming user are currently having network issues. Thus, for a smooth streaming, we need a buffer of dynamic size. To support dynamic buffer, a cooperation between the mobile device and the cellular network is needed. This paper proposes a novel idea—building a cooperation framework to support dynamic playback buffer for improving the performance of streaming applications.

Specifically, we design TANGO, a service that performs real-time data analysis in order to give streaming applications an early notification of impending connectivity degradation, so that the applications can initiate a mitigation action, such as to pre-cache more content. In effect, we turn on its head two operational principles today. First, we enable cooperation between the cellular network and the mobile device through our framework, while current practice does not allow for such cooperation and the mobile device considers the cellular network to be a "dumb pipe". Second, we use this cooperation to allow proactive handling of network condition changes, while the overriding operation mode today is through adaptive bit-rate streaming which reacts to the changing network condition in the hope that the condition will change slowly. Network conditions do change quickly in many scenarios, such as, flash crowds or changing spots with poor signal strength.

An important component of TANGO is user location prediction. Mobility prediction has been studied in several works, with satisfactory results [34], [30], [10]. We implement and evaluate the system using a simple mobility prediction algorithm, based on current and previously connected cell sectors, since this information is readily available, without incurring additional cost, say for GPS measurements. More sophisticated location predictors can be plugged in and potentially result in greater benefits.

We design and implement the two primary components of the system—data analysis and event notification (part of TANGO), and the attendant mitigation action (part of the application). The data analysis considers real data collected by a cellular service provider at the edge elements of its network, the Radio Network Controller (RNC). Information contained in the traces includes actual download/upload rate and current cell sector. This data is finely granular and comprises per mobile device data.

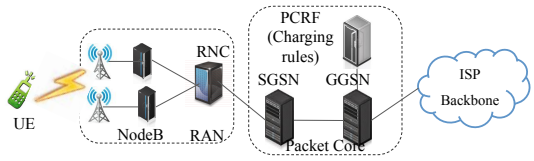

Fig. 1. Architecture of a UMTS data network.

We evaluate TANGO using simulations, using the aforementioned data traces as well as behavior of a real streaming application as the basis. The application used is Pandora, which provides an online radio service, as well a popular javascript-based video streaming client. Effects of stalls are quantified and measured as pause time. We compare TANGO with the baseline approach and DASH [27]. In a simulation with audio streaming users, we found that pre-caching reduces pause time by 13–72%, outperforming DASH by a significant margin, while maintaining higher stream quality.

The rest of the paper is organized as follows. Section II provides background on cellular network. Section III presents the pre-caching service. Section IV presents the experimental evaluation. We follow this with related work and then conclude the paper.

II. BACKGROUND

In this section, we first describe the basics of 3GPP cellular architecture and then provide information about the data set used in our evaluations. The description here applies to 3G/High-Speed Packet Access (HSPA) network, and with some slight modifications to 4G LTE networks as well.

Figure 1 shows the key components of a typical UMTS data network. It consists of 2 major components: the Radio Access Network (RAN) and the Core Network (CN) (or Packet Core). The mobile device, called User Equipment or UE in UMTS terminology, is connected to one or several cell sectors (also referred to as cells) in RAN. This set of cells is called the active set. Only one cell in the active set is actively used for communication at a time. This cell is referred to as the primary cell. A physical base station (called NodeB in 3G and eNodeB in LTE) can have multiple cells, which provide radio resources to UEs for wireless communications. Cellular data traffic from several NodeBs are then passed to the Radio Network Controller (RNC), which manages handovers, and scheduling of wireless resources among the NodeBs under its control. The RNCs connect to Serving GPRS Support Nodes (SGSNs) at the core network. The SGSNs are connected to the external networks, such as the Internet, via Gateway GPRS Support Nodes (GGSNs). When a UE connects to the network, it establishes a Packet Data Protocol (PDP) context which facilitates tunneling of IP traffic from the UE to the peering GGSN using GPRS Tunneling Protocol (GTP) (see [13] for details of the UMTS network).

We evaluate TANGO using real-world cellular traffic data collected at the RNC in a 3G RAN from a tier-1 cellular network carrier. All device and user identifiers are anonymized for our analysis. Details of the dataset are provided in Section IV.

III. DATA PRE-CACHING SERVICE

In this section we describe the data pre-caching service, which notifies applications when connectivity is predicted to be poor in the near future, based on the user's mobility pattern and current load in the network.

A. Overview

A common problem in cellular networks, especially in dense regions is network congestion. A cell's available wireless bandwidth is limited and is shared among users connected to that cell. When the total bandwidth demand exceeds the total available bandwidth, we call that cell *congested*. Adding more capacity in the form of more cell towers, or more number of sectors in an existing antenna, or increasing the transmit/receive power of some antennas are all used today to relieve congestion. However, it is no wonder that these do not completely solve the problem, as seen in the growth of mobile traffic outpacing the growth of network capacity [1], [2].

In addition to congestion, some cells provide persistently lower data rates compared to other cells. This could be due to many reasons such as misconfiguration of antennas, a nearby source of interference, buildings or landscape obstructing the radio signals, or partial hardware failure.

Rather than trying to prevent congestion or the cause of the inferior data rate itself, this service aims to notify the application shortly before the user enters the congested area. Applications can take advantage of this information and respond in various ways—*e.g.*, by pre-caching content that the user is expected to use in the near future (this is the mitigation strategy that we experiment with) or switching to a different carrier or base station within the same carrier. Examples of applications that can use pre-caching as the mitigation action include audio/video streaming (which we use in our experimental evaluation), GPS navigation, and web browsing. These applications have a common property that the future content is known or predictable, to varying extents, and can be downloaded at any time, but in order to conserve bandwidth and/or energy, today's practice is to not predownload content too far in advance. This usual mode of operation works well when network interruptions, if any, are short. However, when the user enters a congested cellular area or a cell with inferior data rate, the amount of buffered content may not be enough to provide uninterrupted user experience. This is known from prior reports, *e.g.*, [22], [12], and we also empirically see this in our experiment where the pause time exceeds 2% of media stream time for a generous fixed buffer size of 10 songs (Figure 3).

Instead of relying on such one-size-fits-all strategy, this service enables applications to utilize different strategies in different network conditions. In general, the pre-caching service can be used whenever a connectivity degradation can be predicted to occur in the near term, such as with lookaheads of a few minutes. This can include situations where the user is entering a tunnel or moving out of the coverage area of her cellular carrier, for example.

In order to provide this service, TANGO needs to be able to predict user location in the near-term future and know the areas that are currently congested as well as cells identified as providing inferior data rates. Congestion information requires real-time network data, as congestion changes dynamically. As we will show in the Section IV-B5, the set of cells providing inferior data rates also change over time. Therefore, the service will require real-time network data from network elements.

Figure 2 shows the overview of the service. During network operation, device location and network load data are collected by the network elements then stored in a database. During offline training, location data from all users in the database is used to train the location prediction model. In the online phase, the application first registers with the service. Then, the device's past and current location is used to continuously predict future location. If a predicted location is currently congested, a pre-caching alert is sent to the application. The application can then decide how much data to pre-cache based on current buffer level, battery level, etc. We discuss location prediction in the next section.

B. Location Prediction

Today, device's location information is available from currently connected cell and GPS measurements. GPS gives more fine-grained location information than current cell does. However, in the common case where GPS is not already being used, using GPS will result in significant energy overhead. Furthermore, for the pre-caching service to be useful, we only need to know coarse-grained information, *i.e.*, which future cells the device will be connected to, since congestion is determined at the granularity of a cell. Therefore, we opt for the current cell as well as past cells as the data source for location prediction.

Both the device and the network are aware of the current primary cell to which the device is currently connected. In our case, we implemented the predictor that uses network data as input, since prediction results will need to be combined with network load data later. Specifically, location predictor's input data consists of the current primary cell as well as the history of past primary cells of the specific device whose location it is trying to predict. Because users with high mobility can move to multiple cells within a short amount of time, instead of predicting a single cell that the user will move to, the predictor estimates the probability that the user will enter a cell C within a specific amount of time, separately for each nearby cell C. As an optimization, the prediction is only needed for cells that are currently congested, or known to be providing inferior data rates.

Symbolically, the predictor works by estimating

$$P(C \in \mathbb{S} | C_t, C_{t-1}, \ldots, C_{t-m}) = \frac{Freq(C \in \mathbb{S}, C_t, C_{t-1}, \ldots, C_{t-m})}{Freq(C_t, C_{t-1}, \ldots, C_{t-m})}$$

from the training data, where C is a future cell, $\mathbb{S}$ is the set of all cells the user will enter in the next u minutes, C_t is the current cell, $C_{t-1}, \ldots, C_{t-m}$ are the history of past cells, and m is the length of cell history. $Freq(C_x, C_{x-1}, \ldots, C_1)$ denotes the frequency (count) of any user visiting cells $C_1, C_2, \ldots, C_x$, in that order. Note that this prediction is about whether the user will enter the cell C in the next u minutes, and not about whether C will be the next cell, and is thus an easier problem to solve. During training, each observed cell sequence of length $m+1$ (for the denominator) and $m+2$ (for the numerator) results in the frequency corresponding to that cell sequence increasing by one. The model's parameters m and u can be set by the user.

Intuitively, the predictor keeps a count of how many times a user enters cell C within u minutes after visiting cells $C_{t-m}, \ldots, C_t$. The counts are kept separately for each sequence $C_{t-m}, \ldots, C_t, C$. This algorithm is extremely simple, and is not designed to improve the state of the art. However, more complex algorithms based on similar input data can be readily plugged in, and the service will potentially benefit from improved accuracy. This benefit is quantified in Section IV-B1.

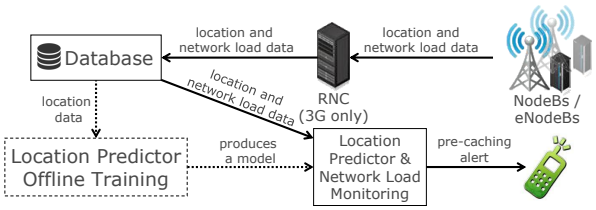

Fig. 2. Overview of pre-caching service in TANGO using network elements that are present in 3G or LTE networks. Dashed lines and dashed boxes represent workflow that only occur during offline training.

As new base stations are added and old ones removed from the network, the location predictor will need to be retrained. Changes in traveling patterns can also cause degraded prediction accuracy. The prediction accuracy should be monitored and another offline training can be triggered once the accuracy falls below a set threshold.

C. Deployment

In this section we discuss the various components that are needed to provide the service, and how they can be implemented in a real cellular network.

The service runs on dedicated servers separate from the existing cellular network components, so as not to impede the time-critical cellular network operations. These servers need access to real-time data about the devices and network load. In 3G networks, network data from base stations is aggregated at the RNCs, where one RNC serves a set of geographically close by base stations. Therefore, it is natural to place the database server at the RNCs, in order to reduce communication latency. In 4G LTE, there are no RNCs; instead, data are collected by the eNodeB's. In this case, the real-time data from the eNodeB's needs to be sent directly to the database server. Since cellular networks can cover a large area, often a whole country, there should be multiple database servers partitioned by location, in order to distribute the load and reduce latency. Each database server is then responsible for storing data corresponding to network operations around it. Since in the online phase the service needs to access data stream as it is generated, the data needs to be *pushed* from the RNCs or eNodeB's rather than having the service *pull* the data as in traditional database systems. For this task we need a streaming data management system, which can run a continuous query on data in motion until the query is explicitly uninstalled.

Having each server provide service for a specific area means that the device needs a way to select the correct server based on its current location. This can be done by having a central "directory server" which directs the device to the correct server each time it uses the service. This directory information can be cached in the device for later use in order to reduce the amount of communication as well as the load on the directory server.

In Section IV, we measure the amount of additional load on the network due to a service in terms of the amount of data involved and the amount of computation needed.

IV. EXPERIMENTS

We evaluate the effectiveness of the data pre-caching service based on real cellular traces from the 3G network of a major US-based cellular network service provider.

A. Data Source

The traces used for all experiments were collected at a RNC of the service provider over the period from February 2013 to November 2014. All device and subscriber identifiers from these traces were anonymized. The dataset, with the device ID, timestamp, and event type as the unique identifier (primary key), contains various performance data and events in the tabular format, with each type of data having its own set of metrics as the columns in addition to the common metrics. The common metrics are the device ID, timestamp, and cells that the device is connected to. Examples of recorded data include download and upload throughput and uplink power, reported at 2 seconds interval, and event-based data items such as, connections and disconnections, handover events, as well as other low-level radio protocol events. There is a second kind of data, which is specific to a cell, and for that the cell ID, timestamp, and event type form the primary key. An example of this type of data is the number of active devices in the cell.

B. Experiments

We designed a simulation experiment mimicking the real-world scenario to evaluate the costs and benefits of data pre-caching in an audio streaming application. The prediction performance as well as overhead of the location predictor is evaluated. Finally, we investigate cells with chronically poor connectivity present in the traces.

1) Reducing stalls in audio streaming: This experiment aims to evaluate if the pre-caching of TANGO helps reduce stalls in an audio streaming application. This would effectively translate to increased capacity of the network for handling audio streaming users, while keeping the quality level the same. The cutoff is provided by the buffering time as a percentage of the total playback time, which for an acceptable quality of use, needs to be kept below a threshold (typically 1-5%, depending on kind of audio and kind of users). We picked streaming audio because online radio services are becoming increasingly popular with Pandora and Spotify having become household names in the US.

We measure the buffering time by running a simulation, with the trace data coming from a real cellular network. An area in the real network is selected, and the cells in the area are simulated. A varying number of emulated audio streaming clients add traffic in addition to existing background traffic with simulated congestions, and the audio pause time is measured. Audio pause time is defined as the total listening duration (say, A), minus the duration where audio is actually played (say, B), minus the initial buffering time (say, C). The quantity $A - B$ gives how long the user has spent waiting for the player to buffer content, thus a smaller value is better for user experience. The quantity C is subtracted from that because that is the initial buffering when the user initiates the request for the audio stream. Presumably, this delay is less annoying to the user than a delay in the midst of listening.

We selected an area in downtown San Francisco, USA to be the simulation area. The area is 1.66 miles long along north-south and 1.91 miles long along east-west. All cells in the area ($>$1000) are included. Each cell is simulated by having a fixed capacity, equal to the maximum theoretical bandwidth of a cell. Existing background traffic as well as users' mobility patterns are taken from the trace between 4-5pm on 17 November, 2014.

The audio streaming users are emulated by mimicking the action of the mobile Pandora application without actually transmitting any data or playing the songs. We experimentally found that Pandora keeps one song in the buffer, in addition to the current song, at all times. When the playback of the current song finishes, it downloads one more song as fast as possible, resulting in a bandwidth spike of a few seconds, and no bandwidth usage afterwards. For each client, the emulator keeps track of how much data has been downloaded for the current song, what the current playback position is, its location (current cell), as well as the audio pause time, and updates this information every second. In place of real traffic, the emulator sends an estimate of how much data the client would request, to the cell proxies. Each cell proxy then sums up the bandwidth demand, and distributes the available bandwidth (after subtracting the background traffic from the total capacity) among the clients proportional to their demands.

In each experiment, six approaches are compared: 1) baseline, 2) DASH, 3) TANGO w/o bit-rate adaptation, 4) TANGO w/o bit-rate adaptation w/ perfect location predictor, 5) TANGO, and 6) TANGO with perfect location predictor. The same default buffer size is used for all six approaches. In baseline, the client always tries to keep the buffer full at all times, and no additional mechanism for reducing stalls is used. In DASH, the future bandwidth is predicted as the harmonic mean of download speed during the last 10 seconds, as recommended by Jiang et al [18]. Three bit-rates are available: 128 Kbps (default), 64 Kbps, and 32 Kbps. The client always pick the highest bit-rate that uses no more than the predicted bandwidth. When predicted bandwidth is lower than 32 Kbps, the lowest bit-rate of 32 Kbps will be chosen. In TANGO, actual location prediction and network load monitoring happen in the same way it would be in the real system. TANGO includes the same bit-rate adaptation mechanism used by DASH. For comparison, we also included variants of TANGO without bit-rate adaptation as well as variants with perfect location predictor. In variants with perfect location predictor, the location predictor is replaced by one that always make correct predictions. This helps quantify the effect the location predictor's accuracy has on the overall benefits of the system.

With TANGO, the location predictor continuously predicts each user's location and monitors cells in the area for congestion. For this experiment, a cell is considered congested if at least $C\%$ of its capacity has been used for the last T seconds. We empirically found the optimal values to be $C = 85\%$ and $T = 30$ seconds. When a user is predicted to be moving to a congested cell in the next 5 minutes, a pre-caching alert is sent to the client emulator. The client emulator pre-caches content by increasing the buffer size from one song to B minutes and download the stream at he default 128 Kbps bit-rate as quickly as possible. The default bit-rate is used in order to prevent degradation of user experience. The optimal buffer size is found to be $B = 30$ minutes.

Each emulated client follows the movement pattern of a real user from the trace. For these experiments, we only

Type	Description
Mobile flash crowds	50 Congestions that move across cells like a user does. Each comes and goes randomly.
Static congestions	Congestions in 20% of the cells that lasts through the whole experiment.
Random congestions	Congestions in 50% of the cells that come randomly and last 0–20 minutes.

TABLE I. DIFFERENT TYPES OF SIMULATED CONGESTIONS USED IN THE EXPERIMENTS

simulate users with some movement between cells, since TANGO will not work for stationary users. Each client's arrival time is uniformly random within the length of the experiment of 1 hour. The session length follows the Weibull distribution with $\lambda = 13$ minutes and $k = 0.52$, as found by Zhang *et al.* to be the best fit of the session lengths of mobile online radio users [33]. This corresponds to the average session length of 24.25 minutes. The relevant characteristics of the songs in the audio stream are the song length and the bit rate. The lengths of the songs come from Spotify's top 50 chart for the week of November 2, 2014 [5].

We mimic congestion in three ways: 1) by adding mobile "flash crowds" to the background traffic, 2) by simulating static congestions in a small fraction of the cells, and 3) by simulating random congestions in a larger fraction of the cells, as shown in Table I.

Each simulated flash crowd follows the movement pattern of a real users randomly picked from the trace. 50 flash crowds are simulated. Their arrival time is uniformly random, and they last as long as the original user stays active. In effect, these flash crowds randomly come and go. While a flash crowd is in a cell, all users connected to that cell cannot download any data. Overall, this causes 1.5% of the cells' operating time to be congested, when summed across all the cells and across all the time points and divided by the total number of time points in the one hour trace.

Static congestions are simulated by randomly picking 20% of all cells as congested. These congested cells does not provide any data to the users connected to them. This effect lasts through the entire 1-hour-long experiment.

Random congestions are similar to static congestions, but the start time and duration are random. This affects 50% of the cells. The start time is picked uniformly at random. The duration ranges from 0 to 20 minutes, also picked uniformly. Overall, this cause 8.3% of the cells' operating time to be congested.

The result of the three simulations is shown in Figure 3. Total audio pause time is measured as a percentage of total playback time. For static congestions and random congestions, there is a critical number of audio streaming users beyond which the patterns change. This critical point, at roughly 7,000 users for static congestions, and 7,500 users for random congestions, corresponds to the point where the capacity of most of the network is reached. For flash crowds, the effect is smoother, so there is no clear-cut critical point, although the general trend is similar. Before the critical point, bit-rate adaptation in both DASH and TANGO barely provides any benefit. This is because there is not much opportunity to use a lower bit-rate, as congestions are sudden, and congested cells provide zero bandwidth. However, TANGO significantly reduces audio pause time compared to DASH and baseline, with TANGO variants with perfect location predictor having a

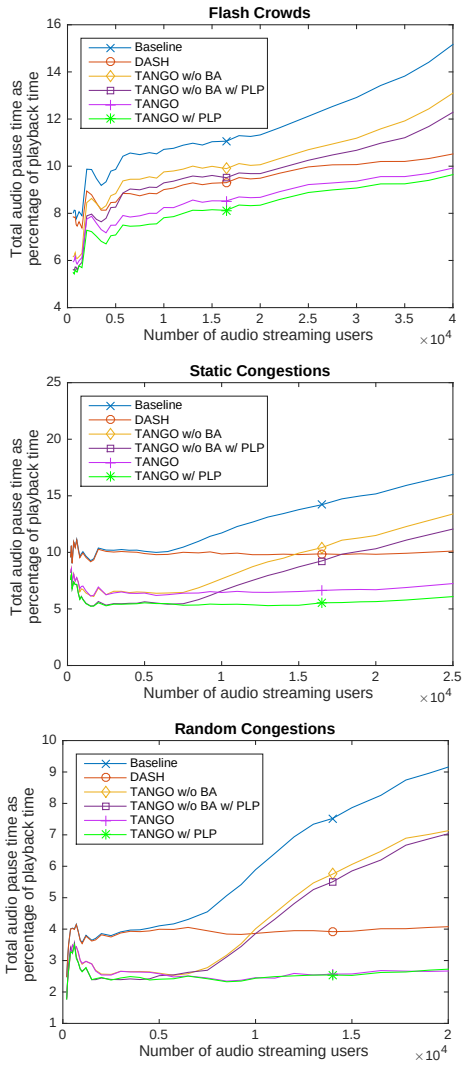

Fig. 3. Audio pause time (shown as percentage of total playback duration) as a function of the number of audio streaming users, with three types of simulated congestions. BA refers to bitrate adaptation, while PLP refers to perfect location predictor. Pre-caching initiated by TANGO significantly reduce pause time due to rebuffering.

slight edge over their counterpart with real (imperfect) location predictor.

After the critical point, however, bit-rate adaptation starts to provide significant benefit. Audio pause time is kept nearly constant while more users are added even after the capacity of the network is reached. This is only possible because the quality of the stream decreases proportionally. TANGO variants without bit-rate adaptation performs better than DASH at lower number of audio streaming users, but quickly performs worse as the number of users increases. TANGO with bit-rate adaptation is able to keep the significant advantage over DASH even with high number of users.

While reducing stalls is important, doing so while significantly lowering the stream's bit-rate is undesirable. We measured the average stream's bit-rate across all users in the simulations, shown in Figure 4. Approaches that do not include bit-rate adaptation always keep the bit-rate constant at 128 Kbps. The general trend is that, the higher the number of users, the lower the average bit-rate. The lower average bit-rate at the lower end of the number of users is likely due to high variance.

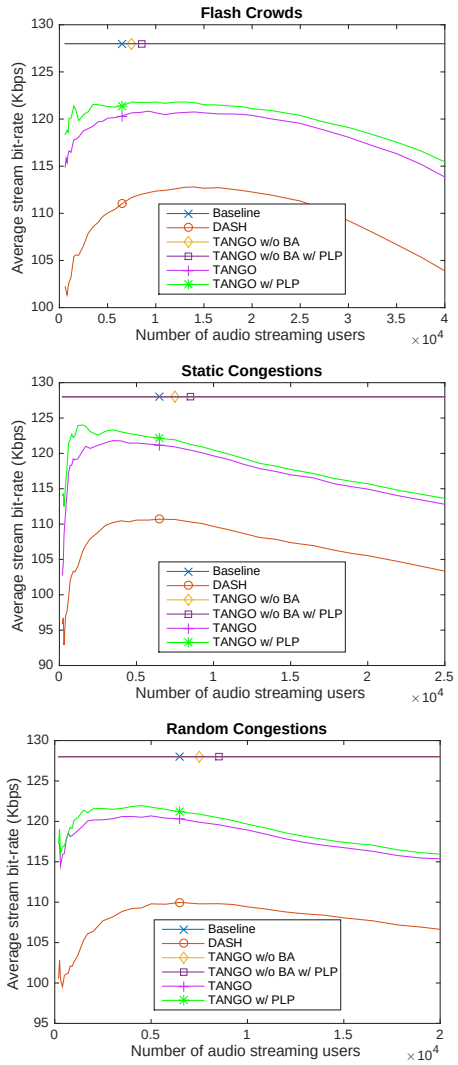

Fig. 4. Average audio stream quality as a function of the number of audio streaming users, with three types of simulated congestions. BA refers to bit-rate adaptation, while PLP refers to perfect location predictor. Approaches that do not include bit-rate adaptation always use the default bit-rate of 128 Kbps.

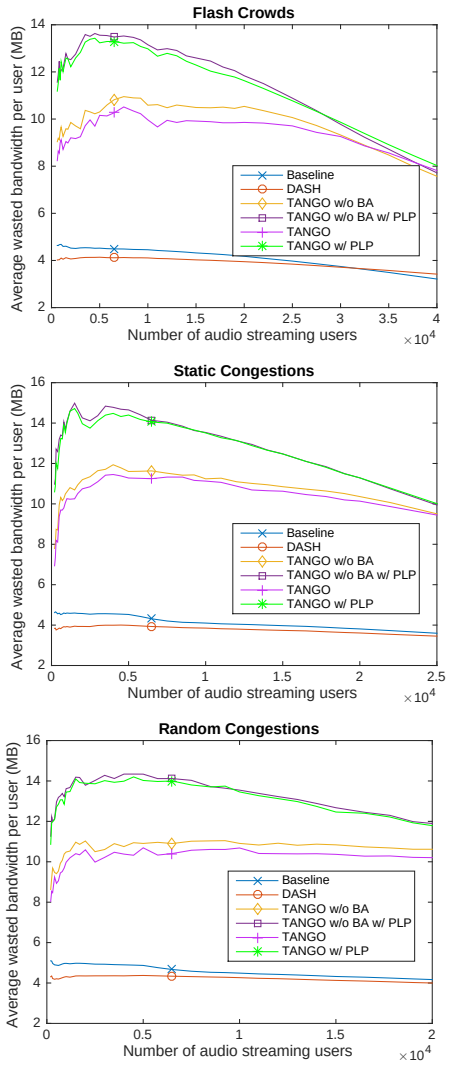

Fig. 5. Average wasted bandwidth per user due to user abandonment (*i.e.*, end of session) for different approaches as a function of the number of audio streaming users. BA refers to bit-rate adaptation, while PLP refers to perfect location predictor.

TANGO's average bit-rate is significantly higher than DASH in all scenarios, at roughly the midpoint between the highest bit-rate of 128 Kbps and DASH's average bit-rate. Looking at the audio pause time together with average bit-rate, we can see that TANGO significantly reduces audio pause time, while maintaining higher stream quality than DASH.

For mobile users, the amount of bandwidth usage usually translates to cost of their service. Therefore, it is important to keep the amount of wasted bandwidth small. Wasted bandwidth is a result of having a non-empty buffer at the end of the user's listening session. We measured the average wasted bandwidth across all users in our experiment, shown in Figure 5.

Both baseline and DASH have a fixed buffer size of one song, so the wasted bandwidth is almost constant. At higher number of audio streaming users the wasted bandwidth decreases slightly due to the fact that some users do not have enough bandwidth to keep the buffer full as well as the lower bit-rate for DASH. In TANGO, the buffer size changes dynamically depending on the predicted connectivity for each user. TANGO produces 2.5–3x the amount of wasted bandwidth compared to baseline and DASH. Bit-rate adaptation only has a small effect on wasted bandwidth. TANGO with perfect location predictor produces more wasted bandwidth than TANGO. Overall, the increase in wasted bandwidth is proportional to the opportunity to pre-cache, which results in reduction of audio pause time. While using TANGO results in more wasted bandwidth, we believe this is a worthy tradeoff for less playback disruption. If desired, lower wasted bandwidth can be achieved by reducing the pre-caching buffer size, at the cost of more playback disruption.

Next, we quantify the effects of varying the default buffer size for each approach in the presence of congestions. Here we fix the number of audio streaming users at 10,000, and vary the default buffer size. Note that TANGO has a separate pre-caching buffer size, which is fixed at 30 minutes. The results are shown in Figure 6. Buffer sizes are specified in terms of the number of songs, in addition to the current song, which is always buffered in its entirety.

The general trend in all case is that larger buffer results in lower pause time. However, it is clear that when the buffer is large, the pause time only depends on whether

bit-rate adaptation is used. This is likely because as buffer becomes larger, overall bandwidth usage increases, and the capacity of the network is reached. Without bit-rate adaptation, the bandwidth required to sustain smooth playback is much higher, so playback is disrupted for users who cannot get the required bandwidth. Nevertheless, such large buffers are not practical, due to the large amount of wasted bandwidth which is inevitable at the end of the user's session. With small buffer, TANGO always perform significantly better than the respective baseline approach (*e.g.*, DASH vs. TANGO). With perfect location predictor, the benefit is larger in flash crowds and static congestions, and almost the same as real location predictor in random congestions.

These results show that without cooperation from the cellular network, an audio streaming player needs to pick a buffer size that gives the best trade-off between low pause time in the presence of congestions (bigger buffer is better), and wasting of energy and bandwidth when the user ends the session. With TANGO, the default buffer size can be kept low, and bigger buffer is used only when an impending connection degradation is predicted.

We find experimentally that when 5,000 audio streaming users are emulated, 64% of the cells are never visited by any emulated user. Of the remaining 36%, the average number of emulated users per cell is 2.03. Thus, there is a non-uniform distribution of users among the cells according to the mobility traces. We also did a parameter sensitivity study of various parameters that determine the behavior of pre-caching and found that non-optimal parameter values still produce close to optimal results. The results are not shown here due to space constraints.

2) Idealized benefit to a single video streaming user: In the previous set of experiments, we showed the benefits of TANGO with audio streaming. In this experiment, we show that TANGO can be equally useful for video streaming applications by reducing the video pause time. The experiment setup consists of a single client moving from uncongested cell to congested cell while playing video that is hosted on an HTTP server. The video streaming application we use is `dash.js` [3] which is a popular javascript-based implementation of MPEG-DASH standard [27]. We chose this player as this is open-source and is actively backed by many leading industry content providers [4]. We modified the player to include pre-caching that gets triggered whenever a congestion alert is sent to the player. The cells are simulated using an HTTP proxy. The proxy's bandwidth is controlled at the server side using the `tc` tool in Linux. Simulated congested cells have bandwidth capacity of 100 Kbps.

In our simulated situation, the player streams a long video that is obtained from the DASH dataset published by Lederer *et al* [19]. The user starts in a non-congested cell and moves to a congested cell after a varied amount of time from 10 seconds to 2 minutes. The time spent in congested cell is fixed at 10 minutes. Thus, this experiment captures an idealized benefit from pre-caching, as a function of lead time.

We compare four approaches in this experiment: 1) baseline, 2) DASH, 3) TANGO w/o bit-rate adaptation, and 4) TANGO. The default buffer size is fixed at 30 seconds for all approaches. In baseline, the video client does not implement bitrate adaptation. In DASH, we follow a typical DASH setting

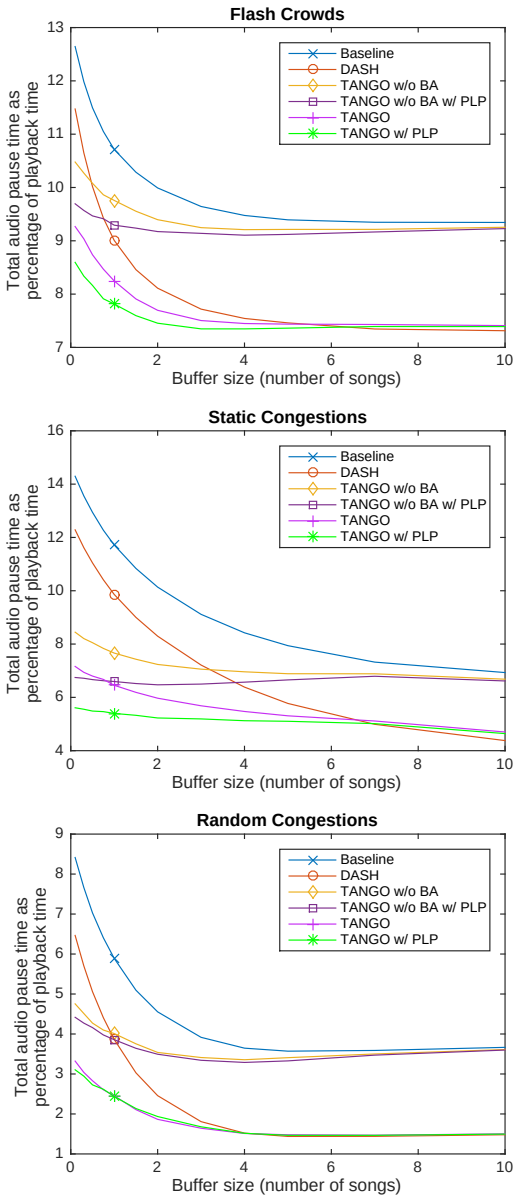

Fig. 6. Audio pause time (shown as percentage of total playback duration) as a function of buffer size (as number of songs), with three types of simulated congestions. BA refers to bitrate adaptation, while PLP refers to perfect location predictor.

with the video available at four different bit-rates: 220, 440, 895, and 1,340 Kbps. In both variants of TANGO, the client will start pre-caching from the beginning of the experiment. The bit-rate adaptation mechanism in TANGO is exactly the same as in DASH.

Figure 7 shows the video pause time as experienced by the client for the four approaches after spending a varied amount of time in the uncongested cell. We can see that both baseline and DASH clients are unable to mitigate the pause time during congestion regardless of time spent in uncongested cell because of the static buffer size. With or without TANGO, bit-rate adaptation helps reduce pause time by lowering the bit-rate while the user is in the congested cell. When lead time is 2 minutes, TANGO is able to pre-cache enough content to last through 10 minutes of congestion. TANGO, which combines

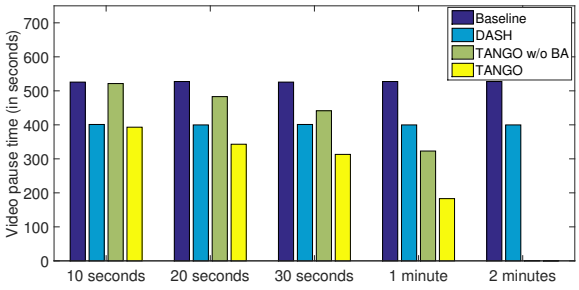

Fig. 7. Video pause time in the simple case where the user is watching one long video and move from a uncongested cell to a congested cell.

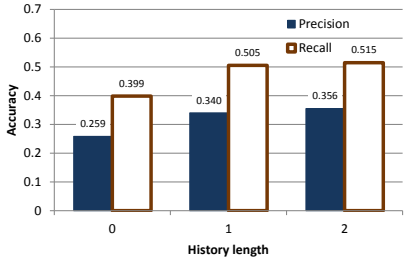

Fig. 8. Location prediction accuracy with varying history length. History length of 0 indicates that only the current primary cell is used.

both pre-caching and adaptation, is most versatile in reducing the pause time as it can prefetch video content when bandwidth is high and also download more video during congestion by reducing the quality which is needed when lead time is small.

3) Location Prediction: Section III describes how the location predictor works. It predicts all the cells that the user will move into in the next 5 minutes, based on the current cell and previously visited cells. There are two important parameters that are known to affect the accuracy of location prediction: the length of history and the probability threshold for generating an alert. This is a micro experiment to investigate how these two parameters affect our location prediction accuracy.

The predictor is trained using data from October 1–20, 2014, and tested on data from October 21–31, 2014. The results for different history lengths are shown in Figure 8. Here, the alert threshold is fixed at 0.15 (if the probability of moving to a congested cell is greater than the alert threshold, then pre-caching is initiated), and the prediction is for the next 5 minutes. Precision is the proportion of all predictions the classifier makes that says the user will move from the current cell to another cell C within the next 5 minutes that are correct predictions. Recall is the proportion of actual user movement to another cell that was predicted correctly. The results show that using only the current primary cell to predict future cells is not adequate. Between history length of 1 and 2, however, the accuracy difference is small. As the model stores counts of how many times a user visits cells $C_1, C_2, \ldots, C_{m+1}$ (where m is the history length) separately for each sequence of cells, the number of parameters needed to be estimated increases exponentially with the history length. Thus, simpler models are generally preferred, so we choose the history length of 1 for our experiments.

We also investigate the effect the alert threshold has on prediction accuracy. For this, we fix the history length to 1. The results are shown in Figure 9. From the results, selecting

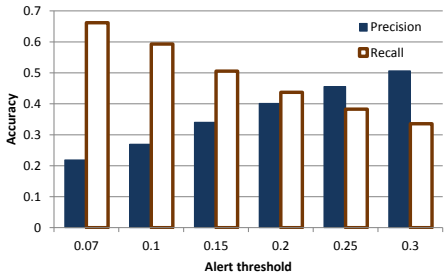

Fig. 9. Location prediction accuracy with varying alert threshold.

the threshold would involve a tradeoff between precision and recall. Therefore, we need to compare the relative cost of false alarm and false negative. A false alarm leads to unnecessary pre-caching, which may lead to wasted bandwidth at the end of a user's listening session. False negative leads to disruption of the stream, which aggravates the user. The relative cost would depend on the user and his service plan (e.g., whether he is on an unlimited data plan). For our experiments, we value less playback disruption higher than bandwidth, and decided to use the threshold of 0.15.

Even with the optimal parameters, the prediction accuracy is still somewhat low. However, there are two specific promising avenues for improvement. The prediction algorithm that we use does not take into account *when* the past cell movement occurred. This information captures the 'speed' of the user, which will let us separate high mobility users from the low mobility users. In addition, in the data traces, there is no direct way of determining cell movements. We use three overlapping events to infer movements of a user from one cell to another — throughput (measured every 2 seconds), handover, and release of a certain radio resource, the last two being event-based. We learn anecdotally that these three put together are still not a complete data source for all cell movements.

4) Overhead: We measured the overhead of the location predictor with respect to the offline training and online prediction. Using a 2.26 GHz Intel Xeon machine, an individual prediction for one user takes only 1 millisecond. On a macro level, one core of the CPU provides enough computation for location prediction for 10.3 million users, each triggering when the user moves from one cell to another. At this number of users, the size of input data stream is 667 KB/s on average. This online overhead scales linearly with the number of users, and the computation can be easily distributed among servers to divide the load. Based on these numbers, a single server is more than adequate for providing location prediction to all users in one RNC.

Offline training involves processing information about all movements between cells within the training period, and therefore takes much longer. Building a model based on 20 days of operation in one RNC, which involves processing approximately 9 gigabytes of data, takes 16.7 minutes. This overhead scales linearly with the number of movements between cells in the training dataset, which is a function of the number of users and their degree of mobility. This offline training can be done in bulk periodically at a central location, or in a distributed manner at the RNCs.

5) Cells with Chronically Poor Connectivity: In addition to cells that are temporarily congested due to a high number of active users, data pre-caching can also be used if we can iden-

tify cells that provide chronically inferior data rates compared to other cells. We find from the traces that the occurrence of chronically underperforming cells is *more common* than transient congestion in the cells. Independent of TANGO, this is a useful characterization of a cellular network.

We identify such cells by comparing their average data rate with those of their neighboring cells, which are more likely to have similar bandwidth demands than cells further away. Specifically, we compute the average data rate in each non-overlapping 30-minute window in the past 7 days of each cell. Then, for each pair of neighboring cells, we perform the *paired t-test*, with the average data rate in a 30-minute window making up a sample. The *p-value threshold* is set to 0.05, with Bonferroni correction. The t-test tells us how likely it is for the difference to be due to chance (*i.e.*, the two cells have the same true distribution of average data rates). We classify cells that have lower data rates (with statistical significance) than 80% of their neighbors as cells with inferior data rate.

We analyze data from period of operation March 1 to March 31, 2014 and found that there are 214 cells with inferior data rates, out of approximately 15,000 cells. Within those inferior cells, there are 874,181 unique active users (those in high power state, with dedicated communication channel). On average, each of these users spends 9.4 minutes (per month) in cells with inferior data rates.

6) Dynamic Changes to Underperforming Cells: In this section, we study how quickly the set of cells identified as providing inferior data rates change over time. The reasoning for this is that, if the cells' quality is static, then there would be no need for real-time communication between the device and the network, since the device can simply pre-download the map of quality of cells and use the location predictor based on local movement information. On the other hand, if the classification changes quickly, it would mean that the models will have to be periodically refreshed.

We analyzed data from multiple RNCs during March 2014 and found that on average, 78% of the cells identified as providing inferior data rates are still identified as such on the next day. This means that in one day, 22% of those cells are no longer providing inferior data rates (while roughly the same amount of cells are newly identified as providing inferior data rates). Thus, if not updated, the classifications will quickly become inaccurate in a matter of a few days. This supports the need for real-time communication between the device and the network, as the device cannot simply cache the database and use it for a long time.

7) Summary Take-aways from Evaluation: We summarize the main take-aways from the experimental evaluation in Table II.

V. Related Work

Adaptive bit-rate streaming has been studied in many works in various aspects. Past measurement studies [7], [15] have pointed out the problems in bit-rate adaptation. Following that, different improvements to bit-rate adaptation have been suggested that adapt either using past TCP throughtput [18], [20] or buffer occupancy [16] or both [32]. TANGO extends current streaming system fundamentally and introduces the idea of buffer adaptation along with bit-rate adaptation using information from the network. An advantage with our approach is that it can be integrated easily with any existing bit-rate adaptation approach.

There are several proposals for managing faults or performance related issues that work either at the network [21], [24], [31] or the device side [8], [25] without any cooperation between them. On the network side, much of the work on fault management in cellular networks has focused on reactive measures, *i.e.*, detection of the failure [21], [24], [11], [14] and then identification of the root cause of the failure [9], [29]. On the device side, Balasubramanian *et al*. [8] use empirical user behavior data regarding mobile web search to schedule data downloads in bursts and reduce energy consumption on the mobile device. TANGO provides the flexibility to implement both reactive and proactive mechanisms (e.g. data pre-caching presented in this paper) on top of it. Furthermore, most of the prior studies use statistical machine learning techniques with very limited amount of data [9], [24], [29]. In contrast, we use a wealth of information from a tier-1 cellular network provider in US. Also, our data set offers fine device and time level granularities for our data modeling and analysis.

The performance of data pre-caching presented in this paper can be further improved by utilizing ideas from several prior studies on handover in cellular networks [26], [17], [28]. Javed *et al*. propose a machine learning framework for predicting handovers [17]. The premise is that handovers cause short-term disruption to application performance, and if an application knows in advanced that a handover is likely to occur in the near future, it can modify its behavior to counter the performance degradation.

User mobility prediction has been studied in several works. Bradley and Rashad propose a prediction technique that take into account different behaviors of users during weekday and weekend [10]. Pathirana *et al.* propose a prediction algorithm that uses data from GPS measurements in an environment where both users and base stations are mobile [23]. Zhang *et al.* use connected cells combined with call records to enhance mobility prediction [34]. These location predictors can be used in place of the simple location predictor used in TANGO, as long as the required input data is available, and prediction lookahead can be set to short term such as 5–10 minutes. Our experiments include results obtained when the location predictor has 100% accuracy as a comparison point.

VI. Conclusions

In this paper, we propose TANGO, a framework for the cellular network and the mobile device to cooperate to improve both network utilization and user experience. We show an instantiation of the framework's pre-caching service, a mechanism for notifying mobile streaming application before the device enters an area with bad connectivity. The streaming application can then adjusts its buffering strategy by increasing the size of the buffer and pre-downloading content into it, so that when connectivity worsens, there is more buffered content available. We evaluate TANGO using real cellular data collected at the edge network element of a major US-based cellular network service provider. Evaluation done using simulated audio streaming application shows that audio pause time is reduced by 13–72% in the presence of different types of congestions, while introducing only a slight decrease in stream's quality and reasonable amount of extra bandwidth usage. We also identify cells with chronically poor connectivity, which can potentially

Audio streaming—pause time	TANGO reduces pause time significantly compared to DASH and baseline. Bit-rate adaptation can almost nullify the effect of having more users in the network.
Audio streaming—stream quality	TANGO always maintains higher stream quality compared to DASH
Audio streaming—wasted bandwidth	TANGO increases bandwidth usage to 2.5–3x in order to pre-cache content
Audio streaming—buffer size	Larger buffer will reduce the effects of congestions more. Baseline and DASH need larger buffer in order to match TANGO's performance.
Runtime overhead	The cost of predicting location is small enough to be usable in practice
Underperforming cells	In the cellular network, 1.4% of the cells are underperforming on any one day; 78% of these will underperform the next day as well

TABLE II. SOME KEY TAKE-AWAYS FROM THE EXPERIMENTAL EVALUATION

be used as a trigger for data pre-caching, and see that there is a fair amount of churn in these cells. As ongoing work, we are furthering the vision of cooperation between mobile devices and the cellular network by building more services that tap into knowledge from both sides.

REFERENCES

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014 - 2019 White Paper. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html.

[2] Cooper's Law. http://www.arraycomm.com/technology/coopers-law/.

[3] Dash-Industry-Forum dash.js player. https://github.com/Dash-Industry-Forum/dash.js/wiki.

[4] Dash-Industry-Forum members. http://dashif.org/members/.

[5] Spotify Charts. http://charts.spotify.com/.

[6] Why Pandora is a Mobile-First Company. http://digiday.com/mobile/why-pandora-is-a-mobile-first-company/.

[7] S. Akhshabi, A. C. Begen, and C. Dovrolis. An experimental evaluation of rate-adaptation algorithms in adaptive streaming over http. In *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, MMSys '11, pages 157–168, New York, NY, USA, 2011. ACM.

[8] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: a measurement study and implications for network applications. In *SIGCOMM '09*, pages 280–293. ACM.

[9] R. Barco, V. Wille, L. Díez, and M. Toril. Learning of model parameters for fault diagnosis in wireless networks. *Wireless Networks*, 16(1):255–271, 2010.

[10] J. Bradley and S. Rashad. Time-based location prediction technique for wireless cellular networks. In *Emerging Trends in Computing, Informatics, Systems Sciences, and Engineering*, pages 937–947. Springer, 2013.

[11] B. Cheung, G. Kumar, and S. A. Rao. Statistical algorithms in fault detection and prediction: Toward a healthier network. *Bell Labs Technical Journal*, 9(4):171–185, 2005.

[12] K. Evensen, T. Kupka, H. Riiser, P. Ni, R. Eg, C. Griwodz, and P. Halvorsen. Adaptive media streaming to mobile devices: challenges, enhancements, and recommendations. *Advances in Multimedia*, 2014:10, 2014.

[13] H. Holma and A. Toskala. *Hsdpa/Hsupa For Umts*. Wiley Online Library, 2006.

[14] C.-Y. Hong, M. Caesar, N. Duffield, and J. Wang. Tiresias: Online anomaly detection for hierarchical operational network data. In *ICDCS '12*, pages 173–182. IEEE.

[15] T.-Y. Huang, N. Handigol, B. Heller, N. McKeown, and R. Johari. Confused, timid, and unstable: Picking a video streaming rate is hard. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 225–238. ACM.

[16] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 187–198. ACM, 2014.

[17] U. Javed, D. Han, R. Caceres, J. Pang, S. Seshan, and A. Varshavsky. Predicting handoffs in 3g networks. *ACM SIGOPS Operating Systems Review*, 45(3):65–70, 2012.

[18] J. Jiang, V. Sekar, and H. Zhang. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, pages 97–108. ACM, 2012.

[19] S. Lederer, C. Müller, and C. Timmerer. Dynamic Adaptive Streaming over HTTP Dataset. In *Proceedings of the 3rd Multimedia Systems Conference*, MMSys '12, pages 89–94, New York, NY, USA, 2012. ACM.

[20] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. Begen, and D. Oran. Probe and adapt: Rate adaptation for http video streaming at scale. *IEEE JSAC*, 2014.

[21] Y. Liu, J. Zhang, M. Jiang, D. Raymer, and J. Strassner. A model-based approach to adding autonomic capabilities to network fault management system. In *NOMS '08*, pages 859–862. IEEE.

[22] O. Oyman and S. Singh. Quality of experience for http adaptive streaming services. *Communications Magazine, IEEE*, 50(4):20–27, 2012.

[23] P. N. Pathirana, A. V. Savkin, and S. Jha. Mobility modelling and trajectory prediction for cellular networks with mobile base stations. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 213–221. ACM, 2003.

[24] S. Rao. Operational fault detection in cellular wireless base-stations. *Network and Service Management, IEEE Transactions on*, 3(2):1–11, 2006.

[25] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan. Bartendr: a practical approach to energy-aware cellular data scheduling. In *MobiCom '10*, pages 85–96. ACM.

[26] S. Seshan, H. Balakrishnan, and R. H. Katz. Handoffs in cellular wireless networks: The daedalus implementation and experience. *Wireless Personal Communications*, 4(2):141–162, 1997.

[27] I. Sodagar. The MPEG-DASH standard for multimedia streaming over the Internet. *IEEE MultiMedia*, 18(4):62–67, April 2011.

[28] T. Sohn, A. Varshavsky, A. LaMarca, M. Y. Chen, T. Choudhury, I. Smith, S. Consolvo, J. Hightower, W. G. Griswold, and E. De Lara. Mobility detection using everyday gsm traces. In *UbiComp '06*, pages 212–224. Springer.

[29] Y. Watanabe, Y. Matsunaga, K. Kobayashi, T. Tonouchi, T. Igakura, S. Nakadai, and K. Kamachi. Utran o&m support system with statistical fault identification and customizable rule sets. In *NOMS '08*, pages 560–573. IEEE.

[30] H. Xiong, D. Zhang, D. Zhang, V. Gauthier, K. Yang, and M. Becker. Mpaas: Mobility prediction as a service in telecom cloud. *Information Systems Frontiers*, 16(1):59–75, 2014.

[31] H. Yan, A. Flavel, Z. Ge, A. Gerber, D. Massey, C. Papadopoulos, H. Shah, and J. Yates. Argus: End-to-end service anomaly detection and localization from an isp's point of view. In *INFOCOM '12*, pages 2756–2760. IEEE.

[32] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli. A control-theoretic approach for dynamic adaptive video streaming over http. *SIGCOMM Comput. Commun. Rev.*, pages 325–338, 2015.

[33] B. Zhang, G. Kreitz, M. Isaksson, J. Ubillos, G. Urdaneta, J. A. Pouwelse, and D. Epema. Understanding user behavior in spotify. In *INFOCOM '13*, pages 220–224. IEEE.

[34] D. Zhang, H. Xiong, L. Yang, and V. Gauither. Nextcell: predicting location using social interplay from cell phone traces. 2013.