Handovers with Forward Admission Control for Adaptive TCP Streaming in LTE-Advanced with Small Cells

Reuven Cohen
Dept. of Computer Science
Technion, Israel

Anna Levin
IBM Haifa Research Labs
Israel

***Abstract*—An important trend in the evolution of cellular networks is the introduction of cost efficient small cells. However, most of these cells will have only wireless connectivity to the backbone. Consequently, handovers will be needed much more frequently and the bandwidth between neighboring cells will become a scarce resource. Both problems are likely to affect one of the most fast growing cellular applications: adaptive TCP video streaming. While the high handover rate is likely to have a negative impact on TCP streaming due to packet loss during handovers, solutions that forward packets from the old cell to the new one must limit the amount of wireless bandwidth they use. This trade-off is addressed in the following paper.**

I. Introduction

One of the most important trends in the evolution of cellular networks is the introduction of cost efficient small cells. The deployment of small cells offers the most promising way to increase the capacity of mobile broadband networks, in order to address the fast growing bandwidth demands due to smartphone streaming and interactive video applications.

Due to cost constraints, and because many sites will be in hard-to-reach locations, most of the eNBs (the LTE base stations) that govern small cells will not have direct (high-speed, terrestrial) backhaul connectivity. An alternative solution is to use the air interface between these eNBs to a set of donor eNBs that have direct backbone connectivity. This concept supported by the LTE-Advanced standard is known as “relaying.”

There are two important consequences to the introduction of small cells that are governed by eNBs with a wireless backhaul connectivity:

1) Handovers will be needed much more frequently.
2) The bandwidth between most of the eNBs and the backbone will become a scarce resource.

Both problems are likely to affect one of the most fast growing cellular applications: adaptive TCP video streaming. While the high handover rate is likely to have a negative impact on TCP streaming due to packet loss during each handover, solutions that forward packets from the old eNB to the new one must use the expensive wireless bandwidth between eNBs and their donors. This trade-off is addressed in the following paper.

The TCP transport protocol is becoming increasingly popular for media streaming applications for several reasons. First, TCP congestion control is essential for guaranteeing network stability. Second, TCP flow control and end-to-end reliability replace Application layer loss recovery. Third, with TCP, the traversal of NAT gateways and firewalls is easier. These issues will prove critical for future wireless networks, which are likely to use wireless technology not only in the last hop but also in the backbone.

This work was partially funded by the EU FP7 SAIL project. This work was conducted in part when Anna Levin was with the Technion.

In adaptive TCP streaming the video is segmented into chunks, and each chunk is requested using a different HTTP GET command. The server encodes each chunk at a bit rate that matches the connection’s most recent throughput. Therefore, higher video quality is obtained for higher throughput. Moreover, for a given throughput, higher quality is obtained when the throughput variance is smaller, because for high variance the client is more likely to encounter an underflow in its playback buffer, and the server is likely to make more errors while estimating the connection throughput.

The LTE network considered in this paper is presented in Figure I. The IP based Core Network (CN) is connected to the Internet via the gateway GW. The base stations (eNBs) are connected to the CN through the Mobility Management Entity (MME) or the Service Gateway (SGW). The eNBs can also directly communicate with each other. The eNBs together with MME need to ensure service continuity to the User Equipment (UE). This is done by means of an effective handover mechanism, which maintains Transport layer sessions during the movement of the UE from one eNB to another.

With the introduction of small cells connected to the backbone using wireless bandwidth, the handover mechanisms must minimize the consumption of the scarce wireless backbone bandwidth, which is used both for backbone connectivity and for packet forwarding. This paper focuses on efficient use of the backbone wireless bandwidth for long-lived TCP connections that are used for media streaming.

The handover procedure in LTE, described in [1] and expanded in [2], [3], works as follows. When the UE moves between two cells, the source eNB (eNB_1) uses measurement reports from that UE to determine the target cell (eNB_2) and asks whether eNB_2 has enough resources to accommodate the UE. After eNB_2 prepares radio resources, eNB_1 commands the UE to handover to the target cell. Then, eNB_1 decides whether to forward or to drop the packets received for the UE after the handover decision has been made, depending on the type of traffic and on the availability of resources:

1) *Drop*: the packets are silently dropped by eNB_1 [1], [2].
2) *Forward*: the packets are forwarded to eNB_2 and then to the UE [1], [3]. Since new packets are routed directly from the sender to the UE through eNB_2, some of the forwarded packets might be received out of order.

Forwarding the packets of a connection from the old eNB to the new one rather than dropping them and leaving the recovery to TCP can significantly improve TCP performance and reduce variance [4], [5], [6]. Therefore, if enough network resources are available for packet forwarding, it is the preferred

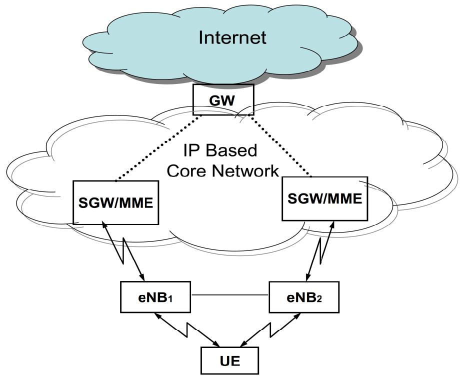


Fig. 1. Handover reference model in LTE network

option for TCP applications. However, the wireless bandwidth for forwarding packets from one eNB to another is a scarce resource. Therefore, the eNBs need to be very selective when deciding whether to drop a packet or to forward it. In this paper we define this optimization problem and seek good solutions.

Generally speaking, we assume that at any given time there is some available bandwidth for the forwarding of packets between any two eNBs of neighboring cells. The old eNB, eNB_1, should consider bandwidth availability when deciding whether to drop or to forward to eNB_2 packets of a certain connection whose UE has just moved to the cell of eNB_2. The decision can be made for each individual connection, or even for each packet, while taking into account bandwidth availability and the TCP's expected profit from forwarding these packets. Even if we consider a simple model, where the cost is equal to the number of packets to be forwarded and the profit is equal to the difference between the throughput of the forward and the drop schemes, the decision is not easy due to its on-line nature.

We consider two different optimization criteria: minimum forwarding cost and maximum throughput. The former is more appropriate for heavily loaded networks, where a limit on the throughput of every connection is attributed to the lack of network resources even if every packet is forwarded. The maximum throughput criterion is more appropriate when the ability of a connection to expand its window before the next handover is mostly limited by losses of the wireless channel. Our solutions for throughput maximization also reduce throughput variance, because they minimize the periods during which *cwnd* is unnecessarily reduced.

The rest of this paper is organized as follows. In Section II we present related work. In Section III we describe the forward and drop handover schemes and discuss their bandwidth cost and profit. In Section IV we present a formal definition of the handover optimization problem and describe our on-line algorithm for this problem. Section V presents a simulation study and Section VI concludes the paper.

II. Related Work

With the increasing capabilities of mobile devices and of the data rates offered by mobile networks, mobile multimedia services over TCP have become popular. Examples of these services are Smooth Streaming by Microsoft [7] and Live HTTP Streaming by Apple [8]. An overview of the multimedia streaming standards for 3GPP networks is presented in [9].

In [10], the authors present an analytical model for evaluating the performance of multimedia streaming over TCP. They explore how various network parameters, such as delay, loss rate, and retransmission timeout, effect TCP streaming throughput. They also show that TCP streaming provides good performance when the available network bandwidth is roughly twice the video bit rate. In [11], the authors address the issue of limiting the latency introduced by TCP. They discuss the importance of low latency to streaming applications and show how such latency can be obtained using dynamic adaptation of the TCP sender's buffer. Unlike [11], we do not focus on the buffers on the path between the TCP source and destination, but on adapting the handover-specific forwarding buffers.

In [12], the authors discuss the importance of TCP-friendliness to adaptive streaming in mobile networks. Using simulations, they show that the ability of a TCP-friendly protocol to dynamically adapt the bit rate of the stream can significantly improve various performance indicators in mobile networks, such as loss rate, delay and buffer space. In our paper we assume that these performance indicators are given and estimate the adaptive streaming performance in terms of overall network throughput. We analyze the behavior of TCP connections at the time of handover and choose the handover scheme that maximizes network throughput given the adaptive behavior of the TCP connections.

The problem we address in this paper is closely related to the problem of call admission control (CAC) in mobile networks. CAC schemes usually look at the handover problem from the perspective of the new eNB and check whether this eNB has enough resources to accept a new UE [13],[14],[15]. Our problem is different because we want to determine the best way to use the bandwidth between two eNBs.

In [16], the problem of finding an optimal routing path between the old eNB and the new one is studied. This path is needed for the packets forwarded at the time of handover and also for the new packets sent by the sender to the UE via the new eNB. The paper presents a solution that minimizes both the signaling for path establishment and the bandwidth for packet transmission. Our paper is different because we assume that the forwarding path is given, and the question is which packets should use it.

The impact of mobility on TCP performance has drawn a lot of attention in recent years. For example, the authors of [2] analyze packet drop scenarios in cellular networks, which are attributed to handovers, to poor wireless link conditions, or to congestion. They suggest incorporating a finite state Markov channel model into the TCP flow control in order to adapt the response of the sender to the real cause of the packet drop. They show that this model indeed improves TCP performance in cellular networks. In [3], the authors analyze the throughput gained by New-Reno when packets are forwarded from eNB_1 to eNB_2. They propose an improvement to the buffer management algorithm in order to prevent overflow, and show that it is better to drop new packets than old ones. In [17], the authors propose a handover scheme that simultaneously transfers packets to both

eNB_1 and eNB_2. When the UE connects to eNB_2, it receives the packets that were not sent by eNB_1. While this solution minimizes packet loss during handover, it is inefficient in terms of bandwidth utilization.

The LTE standard does not support soft handovers. While soft handovers may improve the throughput of VoIP applications [18], it is known to have a negative impact on data applications [19], [20]. In [4] and [6], the authors present performance studies of the standard LTE handover schemes: drop and forward. They show that by forwarding packets from the old eNB to the new one, TCP throughput increases notably because unnecessary timeouts are avoided. In addition, [6] compares the behavior of various TCP flavors during handover. One of the main conclusions is that SACK is the best TCP algorithm for drop, since it can handle multiple packet losses without entering slow-start.

Another topic closely related to this paper is scheduling. In order to use the forwarding bandwidth efficiently, the old eNB should find an on-line schedule of TCP connections whose UE moved to the cell of a neighboring eNB. While this is an on-line problem, an off-line algorithm can be used as a benchmark for estimating on-line algorithm quality. The off-line problem can be defined as "a single machine scheduling problem" whose objective is to minimize the weighted number of "tardy jobs" [21]. The commonly used notation for this problem is $1|r_j|\Sigma w_j U_j$. In [22] it is shown that this problem is NP-hard even if all the connections arrive together at time zero, but it can be solved using a pseudo-polynomial algorithm.

III. Preliminaries

A. Our model

A schematic network model is presented in Figure I. A similar reference model was proposed by [1], [4], [5], [6]. There is a routing path from the GW to eNB_1 and a routing path to eNB_2. Between the GW and the eNBs a tunneling mechanism is used to deliver the data packets to the cell of eNB where the UE is currently located. A link between eNB_1 and eNB_2 can be used for forwarding data between them.

Throughout the paper we consider the standard LTE handover procedure. When the UE moves from the cell of eNB_1 to that of eNB_2, eNB_1 notifies eNB_2 of the upcoming handover. Then eNB_2 prepares radio resources and eNB_1 commands the UE to perform handover to the cell of eNB_2. A control message sent to the GW asks it to direct new packets for this UE to eNB_2. The information gathered by eNB_1 is used to decide whether to drop or to forward to eNB_2 packets arriving after the handover is complete.

B. TCP behavior

In order to estimate the profit from forwarding packets rather than dropping them, we have to understand the behavior of the TCP sender. Throughout the paper we consider TCP SACK, because it was shown to be the best TCP flavor for managing multiple packet losses [6]. Suppose that all $cwnd$ packets pass the GW before it is aware of the handover and arrive at eNB_1 after the UE has moved to eNB_2. Depending on how many packets can be forwarded from eNB_1 to eNB_2, the effect of the handover on the sender can be divided into 4 cases as follows.

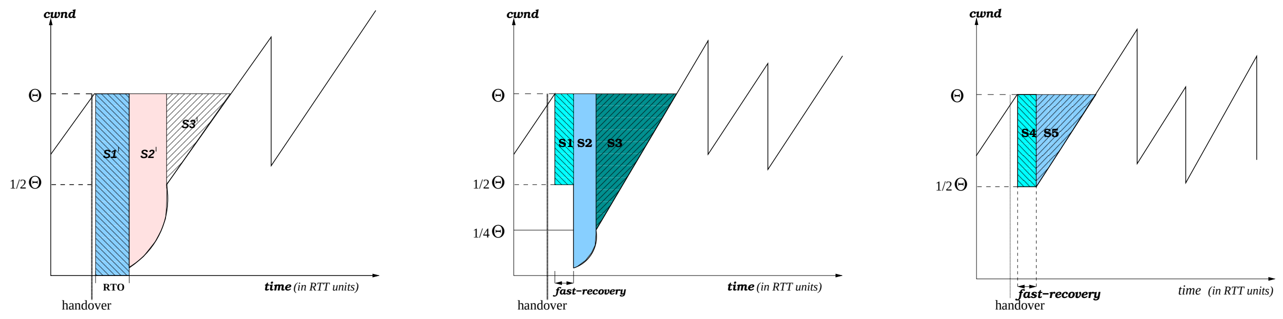
Case (a): All the *cwnd* packets are dropped by eNB_1. In such a case, the connection loses *cwnd* packets. Since the sender does not receive any ACK, it waits for a timeout and then enters slow-start. The *cwnd* curve for this case is shown in Figure 2(a). In this figure, Θ represents the *cwnd* value at the handover time. After a timeout, the *ssthresh* is set to $\frac{\Theta}{2}$. After the connection enters slow-start, the *cwnd* curve grows exponentially until it reaches *ssthresh*. Then, the connection enters congestion-avoidance where *cwnd* grows linearly.

Case (b): Some of the packets are forwarded from eNB_1 to eNB_2, but not enough to avoid a timeout. If less than $\frac{\Theta}{2}$ packets are forwarded and the remaining packets are dropped, the sender receives more than 3 duplicate ACKs and enters fast-recovery. Since there are too many lost packets to proceed with fast-recovery, the connection enters slow-start after a timeout (Figure 2(b)). *Impatient timeout*, recommended for lossy wireless environments especially when *cwnd* is large [23], can also be a problem because it does not allow a TCP sender to stay in fast-recovery longer than the retransmission timeout (RTO). Therefore, if a timeout is to be avoided, the time for resending lost packets must be shorter than RTO. In TCP SACK, resending R lost packets during fast-recovery takes $\log_2 R$ round-trip-times (RTT). Hence, when $\frac{\text{RTO}}{\text{RTT}} < \log_2 R$, the sender enters slow-start after a timeout (Figure 2(b)). When the connection encounters a timeout in Figure 2(b), the threshold is set to $\frac{1}{2} * \frac{\Theta}{2}$. The connection stays in slow-start until *cwnd* reaches $\frac{\Theta}{4}$, and then it enters congestion-avoidance.

Case (c): In this case, enough packets to avoid a timeout are forwarded from eNB_1 to eNB_2. Thus, the sender receives enough ACKs to proceed with fast-recovery. The sender stays in fast-recovery until all dropped packets are retransmitted, and then proceeds to congestion-avoidance. For this case to be applicable, almost all *cwnd* packets should be forwarded. The *cwnd* curve for this case is shown in Figure 2(c). The duration of the fast-recovery stage depends on the forwarding speed of the last packet. We assume that this time is smaller than RTO, because if this is not the case and the impatient timeout option is set, the connection will enter slow-start.

Case (d): In this case, all the packets are forwarded from eNB_1 to eNB_2 in a very short time, such that the delay between the first and the last forwarded packet is smaller than RTT. This connection does not experience any loss or considerable delay avoiding heavy packet reordering typical for case (c). The *cwnd* curve for this case is the regular TCP *cwnd* curve without handover. The connection might experience some delay due to the longer forwarding path, but the overall impact on the throughput is negligible.

From the above discussion it follows that the best performance is achieved in case (d), where the connection is not affected at all, and in case (c), where the connection is only slightly affected. Both cases require eNB_1 to invoke the forwarding scheme. For case (d), all packets must be forwarded, while for case (c) a loss of up to R packets can be tolerated, where $R < 2^{\text{RTO}/\text{RTT}}$ (see case (b)). In order for our algorithms to work with any $\frac{\text{RTO}}{\text{RTT}}$ ratio, and in order to ensure case (d)



(a) with timeout and slow-start (b) with fast-recovery and slow-start (c) without slow-start

Fig. 2. The dynamics of *cwnd* at the time of handover

when possible, we forward all the packets of the connection during handover. However, if the network conditions do not allow forwarding enough packets to avoid a timeout, the *drop* scheme (case (a)), is preferable than the *hybrid* scheme (case (b)), which wastes bandwidth but does not prevent a time-out.

C. Profit estimation

To optimize the benefit of adaptive streaming from a TCP connection, the connection's *cwnd* should be as large as possible and its variance should be minimized. With respect to these two criteria, one can distinguish between the following cases:

(i) The end-to-end bandwidth available to the connection has a bottleneck before the last wireless link.
(ii) The end-to-end bandwidth available to the connection has a bottleneck in the last wireless link, and this bandwidth does not decrease after the handover.
(iii) The end-to-end bandwidth available to the connection has a bottleneck in the last wireless link, and this bandwidth decreases after the handover.

Our scheme aims at improving the performance of the streaming applications in (i) and (ii) by enforcing the *cwnd* to follow cases (c) and (d) of Section III-B. This helps to increase the connection throughput and to decrease its variance.

The bandwidth loss of the *drop* scheme compared to the case where handover does not occur is indicated in Figure 2(a) by the three lined and colored areas. These areas represent:

- The bandwidth loss during the timeout, $S_1' = \Theta * RTO$;
- The bandwidth loss during slow-start. This phase lasts $\log_2 \frac{\Theta}{2}$ RTTs, because *cwnd* is doubled every RTT until it reaches the value of $\frac{1}{2}\Theta$. Thus, we have $S_2' = \Theta * \log_2 \frac{\Theta}{2} - \int_0^{(\log_2 \Theta - 1)} 2^x \, dx = \Theta * \log_2 \frac{\Theta}{2} - \frac{\Theta - 2}{2 \ln 2}$.
- The bandwidth loss during congestion-avoidance until the connection's fair share is reached, $S_3' = \frac{1}{2} * (\frac{1}{2}\Theta)^2 = \frac{\Theta^2}{8}$.

Next, we make similar calculations to estimate the bandwidth loss in the forward scheme. Note that for "very fast" forwarding the bandwidth loss is 0. However, if the forwarding path is not "very fast," case (c) is applicable (Figure 2(c)).

- The bandwidth loss during fast-recovery: in the worst case, the sender needs to send $\frac{\Theta}{2}$ packets during this phase. Therefore, $S_4 \leq \frac{\Theta}{2} * \log_2 \frac{\Theta}{2}$.
- The bandwidth loss during congestion-avoidance until the connection's fair share is reached $S_5 = \frac{1}{2} * (\frac{1}{2}\Theta)^2 = \frac{\Theta^2}{8}$.

To summarize, the profit of a certain connection from using the forward scheme rather than the drop scheme is at least:

$$S_1' + S_2' + S_3' - S_4 - S_5 = \Theta * (RTO - \frac{1}{2}) + \frac{\Theta \log_2 \Theta}{2} - \frac{\Theta - 2}{2 \ln 2} \quad (1)$$

IV. The Selective Forward Algorithm

A. Problem Definition

Consider a TCP connection, TCP$_i$, whose UE moves at time T_i from eNB$_1$ to eNB$_2$. When deciding whether to drop the packets of this connection or to forward them to eNB$_2$, eNB$_1$ must consider:

- The number F$_i$ of packets that eNB$_1$ has to forward in order to avoid a timeout.
- The throughput profit Pr_i from forwarding these packets.
- The bandwidth available for handover at T_i.

As discussed in Section III, F$_i$ can be approximated by Θ, the *cwnd* value at the time of handover, and a lower bound for Pr_i is as given in Eq. 1.

We propose to regulate the bandwidth available on the forwarding path using a token bucket [24]. This scheme consists of two parameters: the token rate r and bucket size B. To implement this scheme, eNB$_1$ maintains a counter for every neighboring cell, which is incremented every $1/r$ time units up to a maximum of B. The values of r and B can be different for different neighboring cells. For equal size packets, one token is used per packet, else one token is used per byte of a packet. When a packet is ready to be forwarded from eNB$_1$ to eNB$_2$, eNB$_1$ checks whether the bucket contains enough tokens. If it does, eNB$_1$ forwards the packet and removes an appropriate number of tokens from the bucket. Otherwise, the packets are queued in a special buffer until more tokens are aggregated. In order to ensure that the buffered packets do not wait too long for a token, the length of a buffer is set to $N = RTO * r$. In the absence of any concrete information, the minimum RTO is set to 1 second [25]. If the buffer is full, the packet is dropped.

We consider the problem of maximizing the overall profit of forwarding TCP connections while not exceeding the bandwidth available for this purpose and refer to this problem as the *TCP Drop or Forward Problem* (TDFP). We propose several solutions (see Table I) and analyze their performance. Drop algorithm simply drops all the packets. The *cwnd* curve for this case is as presented in Figure 2(a). Simple Forward algorithm forwards all the packets that can be admitted by the token bucket. The disadvantage of this algorithm is "fragmentation,"

	Algorithm	Required knowledge	Extra requirements	Action	*cwnd* curve
1	Drop	None	None	Drop packet	Fig. 2(a)
2	Simple Forward	None	Rigid forwarding buffer	Forward packet if buffer space available	Fig. 2(b) or (c)
3	Selective Forward	RTT & RTO	Buffer per connection Elastic forwarding buffer	Accumulate all packets Forward connection if enough bandwidth	Fig. 2(a) or (c)

TABLE I
TDFP ALGORITHMS PROPOSED AND DISCUSSED IN THIS PAPER

namely, that the eNB often forwards several packets from many connections, but not enough for any connection to avoid a timeout. As a result, most of the connections will experience a timeout and the total profit will be small. The Selective Forward algorithm tries to solve this problem.

Before we proceed with the algorithms, let us summarize the set of assumptions that will be used for the rest of the paper:

- The GW becomes aware of a handover and updates its routing tables at least RTT and at most RTO time after the handover takes place.
- All the packets sent by the sender during the first RTT after handover are routed to eNB_1.
- TCP SACK is used, because it was proven to be the best TCP flavor for managing multiple packet loss [6].
- When the handover occurs, the connection is in the congestion-avoidance phase and the profit from forwarding is measured according to Eq. 1.

B. The Algorithm

The Selective Forward algorithm forwards the packets of a certain connection if it predicts that there is enough bandwidth for forwarding at least F_i packets of this connection, which is the minimum in order to prevent a timeout. The number of packets that can be admitted for forwarding by eNB_1 at time t is the sum of the number $B[t]$ of tokens accumulated in the bucket and the available space in the forwarding buffer. This space is represented by $N - Q[t]$, where $Q[t]$ is buffer occupancy at t and N is the maximum buffer size. If $N - Q[t] + B[t] < \text{F}_i$, all packets of the considered connection are dropped by eNB_1.

To forward all of the packets received by eNB_1, eNB_1 needs to accumulate them in a separate buffer and count them. In order to decide when to stop accumulating packets and to make a forward/drop decision, eNB_1 estimates the connection RTT, e.g., using the algorithm proposed by [26]. According to this algorithm, eNB_1 records the sequence number and the transmission time of every packet transmitted to the UE. Then, eNB_1 matches the ACKs with the recorded sequence numbers. The same operation is done for the packets sent by the UE and both directions estimations are summed up into an RTT sample. Additional samples are added using a moving average.

To summarize, Selective Forward Algorithm is as follows:

Algorithm. *(The Selective Forward Algorithm)*

G-1 *At T_i, when the UE moves to the cell of eNB_2, eNB_1 starts accumulating packets of the connection.*

G-2 *At $T_i + RTT_i$, where RTT_i is the estimated RTT for the connection i, eNB_1 estimates the value of F_i as the number of packets accumulated by that time.*

G-3 *If $N - Q[T_i + RTT_i] + B[T_i + RTT_i] \geq F_i$, all the F_i packets are forwarded. Otherwise, they are dropped.*

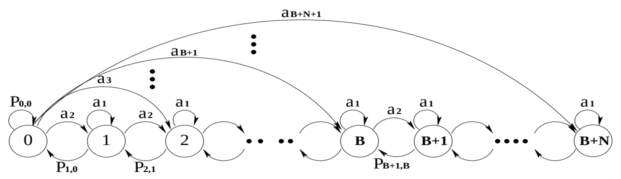


Fig. 3. A Markov chain model for the Selective Forward algorithm

C. Algorithm Analysis

We now present a stochastic analysis of Algorithm 3 in order to estimate its expected profit for the whole network. Recall that the profit for each connection is as indicated by Eq. 1. Suppose that connections move from the cell of eNB_1 to the cell of eNB_2 according to a Poisson process with a rate λ. Let the number F_i of packets to be forwarded be an exponentially distributed random variable whose expected value is F. We define a discrete-time Markov chain that describes the stochastic behavior of the eNB's forwarding resources (Figure 3). State $i \in \{0, 1, ..., B\}$ corresponds to having $B - i$ tokens in the bucket and no waiting packets in the forwarding buffer. State $i \in \{B+1, B+2, ..., B+N\}$ corresponds to having $i - B$ waiting packets in a forwarding buffer and no token in the bucket. A state transition occurs at least once every $1/r$ time units, when a new token is created. This time interval is short enough to have no more than one handover. The admission of a new handover that requires forwarding of j packets is represented by a forward a_j arrow from state i to state $i+j-1$. A backward arrow represents the creation of a new token, which is added to the bucket or used for packet transmission. The transition probabilities for this chain are:

$$\text{P}_{0i} = \begin{cases} 1 - \Sigma_{k=2}^{B+N+1} a_k & \text{for } i = 0 \\ a_{i+1} & \text{for } 1 \leq i \leq B+N \end{cases} \tag{2}$$

and, for $j \geq 1$,

$$\text{P}_{ji} = \begin{cases} 1 - \Sigma_{k=1}^{B+N-j+1} a_k & \text{for } i = j-1 \\ a_{i-j+1} & \text{for } j \leq i \leq B+N \\ 0 & \text{otherwise.} \end{cases}$$

When the system is in state $i = N + B$, it can accept a new connection only with a single packet to be forwarded, with the profit Pr_1. Thus, the expected profit for state $i = N + B$ is $E_{N+B} = a_1 * \text{Pr}_1$. Generalizing this idea to the other states:

$$E_i = \begin{cases} a_1 * \text{Pr}_1 & \text{for } i = B+N \\ \sum_{k=1}^{B+N-i+1} a_k \text{Pr}_k & \text{for } 0 \leq i < B+N, \end{cases} \tag{3}$$

and the profit of a whole system is:

$$E(\text{Profit of Selective Forward Algorithm}) = \sum_{i=0}^{B+N} p_i * E_i, \tag{4}$$

where p_i is the steady state probability of state i. To find the value of p_i, we solve the following set of equations:

$$\begin{cases} p_1 = p_0 * \frac{\sum_{k=2}^{B+N+1} a_k}{1-\sum_{k=1}^{B+N} a_k} \\ p_{n+1} = \frac{p_n - \sum_{k=0}^{n} p_k a_{n-k+1}}{1-\sum_{k=1}^{B+N-n} a_k} \quad \forall 1 \leq n < B+N \\ \sum_{n=0}^{B+N} p_n = 1 \end{cases}$$

and get

$$\begin{cases} p_0 = \frac{1}{\sum_{k=0}^{B+N} C_k} \\ p_i = C_i * p_0 \qquad \forall 0 < i \leq B+N, \end{cases}$$

where

$$\begin{cases} C_0 = 1 \\ C_1 = \frac{\sum_{k=2}^{B+N+1} a_k}{1-\sum_{k=1}^{B+N} a_k} \\ C_{n+1} = \frac{C_n - \sum_{k=0}^{n} C_k a_{n+1-k}}{1-\sum_{k=1}^{B+N-n} a_k} \quad \forall 1 \leq n < B+N. \end{cases}$$

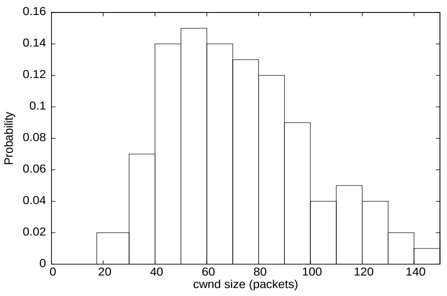
V. Simulation Study

We now present a simulation study using ns-2. We consider a large LTE network with many small cells. We start by considering 20 UE nodes. Our LTE-advanced network model consists of multihop wireless paths in CN, where packet loss in the wireless links plays a major role. We assume that the packet loss rate seen by TCP is between 10^{-4} and 10^{-6} for the whole path. The fair share of the bandwidth available for each connection is 50Mbps. However, the connections utilize only 10% of the available bandwidth because the relatively high packet loss rate prevents them from gaining their full fair share. The propagation delay from the sender to the UE is 110ms.

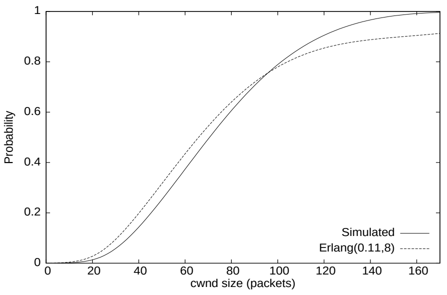
The time spent by a connection in each cell is exponentially distributed with an average of 20s. This short time is justified by the small size of the cells in LTE-advanced networks. Thus the interarrival time between handovers is exponentially distributed with an average of 0.05s. We use the average connection size and interarrival time to calculate the maximum bandwidth needed for forwarding all the packets of all the connections that experience handovers. The bandwidth available for forwarding varies between 0 and 100% of the maximum bandwidth.

During the simulations we collected statistical data for the *cwnd* size distribution, which appears to be very close to the Erlang distribution. Figure 4(a) presents the histogram of the $cwnd$ distribution for the 10^{-4} packet loss rate, and Figure 4(b) shows the similarity to Erlang(0.11, 8). We found the Erlang's shape and λ using the maximum likelihood estimation [27].

Figure 5 presents a comparison between the profit obtained through the simulations and the profit obtained through the theoretical analysis of Selective Forward (Eq. 4). The x-axis represents the actual bandwidth available for forwarding as a fraction of the bandwidth required in order to forward all the packets of all the connections. The y-axis represents the obtained profit as a fraction of the maximum profit, which is obtained by forwarding all of the packets. The $cwnd$ values of the connections are distributed according to the Erlang distribution from Figure 4(b). The upper bound of the profit is calculated while assuming that forwarding is very fast and the connections avoid fast-recovery (case(d) in Section III-B).



(a) *cwnd* size histogram



(b) *cwnd* cumulative distribution

Fig. 4. *cwnd* statistics at the time of handover

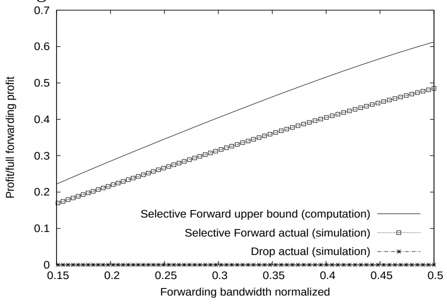


Fig. 5. Simulated profit and its theoretic upper bound

We see in Figure 5 that the profit obtained through simulations is relatively close to the upper bound computed by our analysis. The gap between the lines is explained by inability of the real connections to avoid fast-recovery, especially when the normalized forwarding bandwidth is less than 0.5. The graph also shows that using Selective Forward is profitable in terms of the overall throughput. Even when the normalized forwarding bandwidth is only 0.5, which means that only half of the required packets can be forwarded, the forwarding profit gain reaches 50% of the maximum possible value.

Figure 6 compares the throughput obtained by the 3 algorithms from Table I. It shows the average throughput achieved by a TCP connection in the new cell normalized to the throughput obtained by the Drop algorithm. As before, the x-axis represents the normalized bandwidth available for forwarding. In Figure 6, 1/3 of the connections have lower transmission loss rate, and their *cwnd* is twice as big as the *cwnd* of the remaining 2/3 of the connections. Recall that the Simple Forward algorithm forwards all packets of the connections that experience handover without taking into consideration other connections or checking availability of the forwarding resources. The graph shows that the performance of the Simple Forward is even worse than the performance of Drop for small bandwidth.This

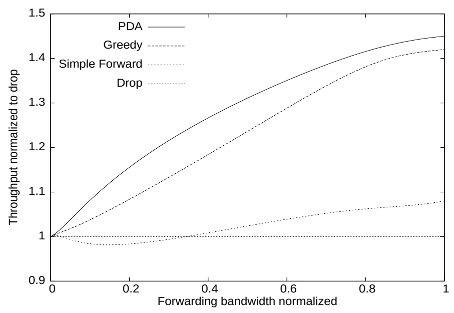


Fig. 6. Throughput in a new cell for various algorithms

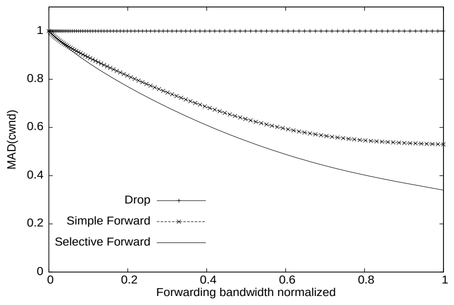


Fig. 7. *cwnd* MAD upon handover (non-equal connections)

somewhat surprising observation can be explained as follows:

- When the packets of different connections arrive interchangeably, forwarding a little bit of everything may result in fast-recovery followed by slow-start (see Figure 2(b)).
- When packets are added to the forwarding buffer in a burst, many are likely to get lost.

As stated earlier, not only is high throughput important for adaptive streaming, but also small *cwnd* variance. Figure 7 shows the Mean Absolute Deviation (MAD) of the *cwnd* after handover normalized to the mean *cwnd* before handover as a function of the normalized available bandwidth: $MAD = \frac{1}{N}\sum_{i=1}{}^{N}\left|cwnd_i^A - cwnd_i^B\right|$, where $cwnd_i^B$ and $cwnd_i^A$ are a pair of *cwnd* values before and after handover respectively.

We can see that the normalized MAD in Drop is not affected by the available bandwidth, and that it is approximately equal to 1, which is the maximum normalized distance between $cwnd_i^B$ and $cwnd_i^A$. For Selective Forward, when the forwarding bandwidth is small, normalized MAD is also close to 1, but when the bandwidth grows, MAD becomes closer to 0, which indicates that $cwnd_i^B$ and $cwnd_i^A$ are almost equal. For Simple Forward the MAD is close to 0.5, because most of the connections enter fast-recovery reducing their *cwnd* by half even when the normalized available bandwidth is close to 1. This is due to the small, rigid buffer used by this algorithm, which is unable to accommodate all the packets that should be forwarded.

The value of MAD does not reach 0 for two reasons. First, there is still a small packet loss rate in the network, which reduces *cwnd*. Second, when forwarding is not very fast, some of the connections experience packet reordering, which leads to fast-recovery and *cwnd* reduction, as in Figure 2(c). This can be avoided by shortening waiting time in the forwarding queue.

VI. Conclusions

We presented efficient handover schemes to maximize the performance of adaptive streaming TCP in LTE networks when the bandwidth between the eNBs is a scarce resource. Our handover schemes seek to maintain the TCP throughput while minimizing the traffic exchange between eNBs. We proposed 3 handover algorithms and analyzed their performance. The first algorithm drops all the packets received by the old eNB after the UE has moved to a new cell. The second algorithm forwards all these packets to the new eNB. The third algorithm uses an elastic forwarding buffer and makes an on-line decision which packets to drop and which to forward. We showed that the best performance is obtained by the third algorithm, and that this algorithm not only maximizes the total throughput but also minimizes its variance during handover, thus improving users' experience of streaming applications.

References

[1] "Long term evolution (LTE): A technical overview," *Motorola*, 2007.
[2] N. Fei Hu; Sharma, "The quantitative analysis of TCP congestion control algorithm in third-generation cellular networks based on FSMC loss model and its performance enhancement," *INFOCOM 2002.*
[3] C.-T. Chou and K. G. Shin, "Smooth handoff with enhanced packet buffering-and-forwarding in wireless/mobile networks," *Wireless Networks*, vol. 13, no. 3, pp. 285 – 297, 2007.
[4] A. Racz and et al, "Handover performance in 3gpp long term evolution (LTE) systems," in *Mobile and Wireless Communications Summit, 2007*.
[5] D. Pacifico and et al, "Improving TCP performance during the intra LTE handover," in *GLOBECOM 2009*, Dec 2009.
[6] R. Cohen and A. Levin, "To drop or not to drop: On the impact of handovers on TCP performance," *INFOCOM MOVE 2008*.
[7] A. Zambelli, "IIS smooth streaming technical overview." [Online]. Available: http://www.microsoft.com/downloads/
[8] R. Pantos, "HTTP live streaming." [Online]. Available: tools.ietf.org/html/draft-pantos-http-live-streaming
[9] F. Gabin and et al, "3GPP mobile multimedia streaming standards [standards in a nutshell]," *Signal Processing Magazine*, vol. 27, 2010.
[10] B. Wang and et al, "Multimedia streaming via TCP: an analytic performance study," in *ACM international conference on Multimedia*, 2004.
[11] A. Goel, C. Krasic, and J. Walpole, "Low-latency adaptive streaming over TCP," *ACM TOMCCAP*, vol. 4, August 2008.
[12] K. Tappayuthpijarn and et al, "Adaptive video streaming over a mobile network with TCP-friendly rate control," *IWCMC*, pp. 1325–1329, 2009.
[13] C. Oliveira, J. B. Kim, and T. Suda, "An adaptive bandwidth reservation scheme for high-speed multimedia wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 16, pp. 858–874, Aug 1998.
[14] W. Ibrahim, J. W. Chinneck, and S. Periyalwar, "A QoS-based charging and resource allocation framework for next generation wireless networks," *Wireless Communications and Mobile Computing*, pp. 895–906, 2003.
[15] M. Ahmed, "Call admission control in wireless networks: a comprehensive survey," *IEEE Comm. Surveys and Tutorials*, vol. 7, pp. 49–68, 2005.
[16] Y. Bejerano and et al, "Efficient handoff rerouting algorithms: a competitive on-line algorithmic approach," *IEEE/ACM Trans. Netw.*, 2002.
[17] M. Saito and et al, "IP packet loss prevention scheme with bicast and forwarding for handover in mobile communications," *IEICE Trans.s 2008*.
[18] T. Lee and et al, "A new soft handover mechanism using DCHs in 3GPP HSDPA networks," *Journal of Networks*, vol. 4, pp. 184–191, 2009.
[19] "Mobile WiMax -Part I: A technical overview and performance evaluation," *WiMax Forum*, Aug. 2006.
[20] "Mobile WiMax -Part II: A comparative analysis," *WiMax Forum*, 2006.
[21] P. Brucker, *Scheduling Algorithms*. SpringerVerlag, 2004.
[22] E. L. Lawler, "A dynamic programming algorithm for preemptive scheduling of a single machine to minimize the number of late jobs," *Annals of Operations Research*, vol. 26, no. 1, pp. 125–133, 1990.
[23] X. Li and et al, "TCP performance in IEEE 802.11-based ad hoc networks with multiple wireless lossy links," *Trans. on Mobile Comp.*, 2007.
[24] I. Cidon and I. Gopal, "PARIS: An approach to integrated high-speed private networks." *INT. J. DIGITAL ANALOG CABLED SYST.*, 1988.
[25] V. Paxson and M. Allman, "Computing TCP's retransmission timer," in *IETF RFC 2988*, 2000.
[26] J. But, U. Keller, and G. Armitage, "Passive TCP stream estimation of rtt and jitter parameters," in *LCN '05*.
[27] G. K. Miller, "Maximum likelihood estimation for the Erlang integer parameter," *Statistics and Probability Letters*, 1999.