University of Würzburg
Institute of Computer Science
Research Report Series

Performance Analysis of the Trade-off Between Signalling Load and Power Consumption for Popular Smartphone Apps in 3G Networks

Christian Schwartz, Tobias Hossfeld,
Frank Lehrieder, Phuoc Tran-Gia

Report No. 485 January 2013

University of Würzburg, Germany
Institute of Computer Science
Chair of Communication Networks
Am Hubland, D-97074 Würzburg, Germany
`prename.name@uni-wuerzburg.de`

Performance Analysis of the Trade-off Between Signalling Load and Power Consumption for Popular Smartphone Apps in 3G Networks

Christian Schwartz, Tobias Hossfeld,
Frank Lehrieder, Phuoc Tran-Gia
University of Würzburg, Germany
Institute of Computer Science
Chair of Communication Networks
Am Hubland, D-97074 Würzburg, Germany
prename.name@uni-wuerzburg.de

Abstract

The popularity of smartphones and mobile applications has experienced a considerable growth during recent years. As a consequence, efficient resource management mechanisms are required to cope with the increasing demands and user expectations. Since smartphones have only very limited energy resources, battery efficiency is one of the determining factors for a good user experience. Therefore, some smartphones tear down connections to the mobile network soon after a completed data transmission to reduce the power consumption of their transmission unit. However, frequent connection re-establishments caused by apps which send or receive small amounts of data often lead to a heavy signalling load within the mobile network.

The investigation of this trade-off between energy consumption at the smartphone and the generated signalling traffic in the mobile network is one of the major contributions of this paper. We explain that this trade-off can be controlled by the connection release timeout and study the impact of this parameter for a number popular apps that cover a wide range of traffic characteristics in terms of bandwidth requirements and resulting signalling traffic. Finally, we study the impact of the timer settings on QoE for web traffic. This is an important aspect since connection establishments do not only lead to signalling traffic, but they also increase the load time of web pages.

Contents

1 Introduction

Together with the wide-spread usage of smartphones in today's UMTS networks, the popularity of smartphone apps has seen a tremendous growth during the last years [1, 2]. The resulting traffic is expected to exceed half of global mobile data traffic in the next years [3]. One of the major reasons for this phenomenon is that smartphones are very convenient for users to stay always connected to the Internet. In turn, this has led developers of smartphone apps to the assumption of continuous Internet connectivity. Therefore, many apps such as social network clients, weather forecasts or instant messengers update their status frequently, which raises a number of problems – in the mobile network as well as on the smartphones.

In contrast to desktops or laptops, smartphones are equipped only with limited battery. Since established connections from the smartphone to the mobile network consume a large amount of energy, some smartphones close these connections soon after the data transmission is finished, i.e., after a very short period of no traffic activity, which is controlled by an inactivity timer. This saves energy and prevents battery drain caused by established but unused connections. However, it might also degrade the user experience since the connection start-up delay is in the order of a few seconds. Therefore, users might get annoyed, for example if the load time of every web page in their browser is increased. Hence, two oppositional effects impact the Quality of Experience (QoE) perceived by the user: lower energy consumption and short page load times – a trade-off that we investigate thoroughly in this paper.

From a mobile operator's point of view, each change between connected and disconnected states of the smartphone causes signalling in the network and excessive signalling load has lead to severe problems in the recent past. For example, a large mobile network in Japan suffered from an outage of several hours in January 2012 [4, 5]. The operator of this network mentioned a large number of small control messages of certain popular apps (such as keep-alive messages or buddy list updates in VoIP apps) as a probable cause for this outage [5]. Therefore, we argue that not an overload in data traffic but the high number of establishments and tear-downs of wireless connections has brought down the network. As a consequence, the network operator may adjust the timer setting accordingly. A longer timer setting can reduce signalling load and resource costs of the mobile operator, however, at the cost of energy consumption of the user device. Hence, an additional trade-off exists between battery efficiency on smartphones and signalling load in the mobile network. The proprietary fast dormancy mode as implemented by smartphones (i.e. smartphones tear down the connection earlier than advised by the network) additionally affects the signalling load in the network and is also investigated in this paper.

A reason for this signalling storm is the following. UMTS networks are designed to provide wireless, high bandwidth Internet access for mobile users, for example for video telephony. Therefore, high load in terms of data traffic was carefully considered during the design of such networks. However, some currently popular apps such as Aupeo radio streaming or Skype VoIP apps load the mobile network all the time, even if they are only running in the background. They send update and keep-alive messages every few seconds, which is problematic for today's UMTS networks. The reason is that the mobile network usually tears down wireless connections to a User Equipment (UE) after an inactivity timeout of a few seconds, i.e., when no data was transmitted for this short time. If the time between two such keep-alive messages is slightly above the connection timeout of the mobile network, the wireless connections between the UE and the mobile network are established and torn down every few seconds. This is an issue that UMTS networks have not been designed for.

The *contributions of this paper* are the following. First, we study the trade-off between energy consumption at the smartphone and the generated signalling traffic in the mobile network. From the user's point of view, battery efficiency is one of the determining factors for a good QoE due to very limited energy capacity of smartphones [6]. Thereby, we analyse the impact of the inactivity timer as well as of the fast dormancy mode for exemplary smartphone apps based on measurements in a public 3G network. These apps are selected to cover a wide range of traffic characteristics in terms of bandwidth requirements and resulting signalling traffic. In particular, we consider background applications (like Twitter or Skype in passive mode), bandwidth intensive (like Aupeo radio streaming), and interactive applications (like Angry Birds). Then, we answer the question whether a mobile operator is able to find optimal values for the inactivity timer. Further, we see which apps (and which traffic characteristics) make the smartphone users and the mobile operators angry, respectively. Furthermore, we discuss new paradigms for the design of mechanisms which try to resolve such conflicts, namely, economic traffic management [7] and design for tussle [8, 9]. Finally, we study the impact of the timer settings on QoE for web browsing based on existing models for web browsing [10]. This is an important aspect since connection establishments do not only lead to signalling traffic, but they also increase the load time of web pages which is the key influence factor on web QoE.

The *remainder of this paper* is structured as follows. Section 2 provides the background on UMTS networks and the radio resource control protocol used for connection establishment and tear-down. Related work on measurement studies of relevant RRC parameters like the inactivity timer is reviewed. Further, existing optimisation approaches of the resource consumption are revisited. Then, energy consumption as key QoE influence factor of smartphone users is considered. Section 3 describes the measurement setup and the algorithm to infer (connectivity and energy) state transitions of the smartphone from measured IP packets. Then, we calculate signalling frequency and power consumption based on the state transitions. Section 4 presents the numerical results of our analysis. The traffic of the four popular smartphone apps are characterized. Afterwards, we compare the signalling frequency and power consumption depending on the network configuration. Finally, we study the influence of network parameters on QoE for web browsing based on page load times. Section 5 concludes this work with an outlook on open challenges how to optimise mobile networks without annoying users or operators.

2 Background and Related Work

This section provides background information on the structure of UMTS networks and their components. In addition, it explains the Radio Resource Control (RRC) protocol, which is used for the allocation of wireless transmission resources. Afterwards, it reviews related work on measurement studies of RRC parameters in real networks and on optimisation approaches of such resources, both in the energy and the wireless domain. Finally, we discuss the impact of energy consumption on mobile devices such as smartphones on the QoE of the end user.

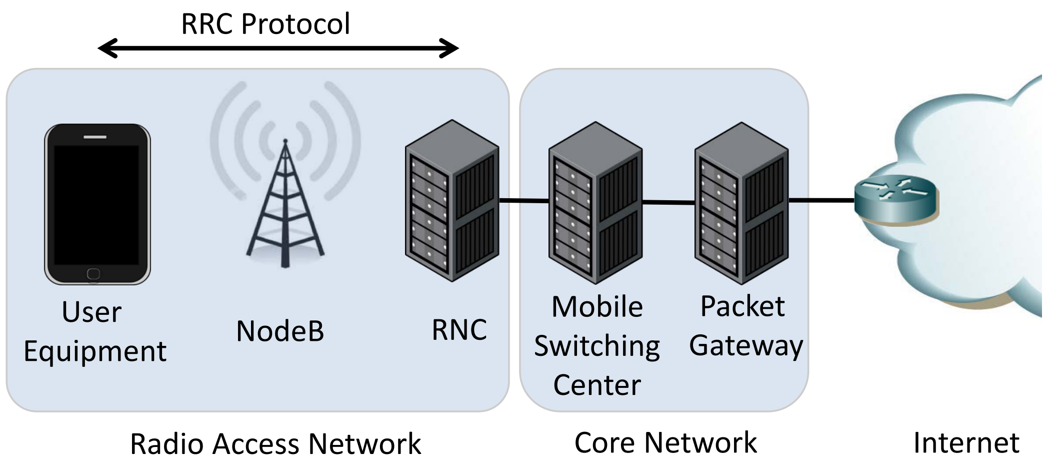


Figure 1: Basic structure of a UMTS network for data calls.

2.1 UMTS Networks and the RRC Protocol

UMTS networks consist of the UEs, the Radio Access Network (RAN), and the Core Network (CN). Their basic structure for packet-switched connections is illustrated in Figure 1. UEs are usually cell phones or computers connected using datacards. The RAN connects the UEs to the CN, which is connected to the Internet. The RAN contains the NodeBs which the UEs are connected to using the air interface. The NodeBs in turn are connected to Radio Network Controllers (RNCs), which among other tasks are responsible for radio resource control using the RRC protocol [11].

For resource management reasons a UE may be in one of several RRC states, where each state allows the UE a different level of connectivity to the RAN. The RRC protocol specifies 5 states for the connection between the UE and the RAN: Idle, URA_PCH, CELL_PCH, Forward Access Channel (FACH), and Dedicated Transport Channel (DCH).

In the Idle state, the cell is notified of the UEs presence, but the UE can neither send nor receive data. The FACH and DCH states allow communication with the RAN, where the DCH state allows for larger bandwidth at the cost of a higher power consumption. For simplicity reasons, we neglect URA_PCH and CELL_PCH in this study. While URA_PCH plays only a role in scenarios of high mobility, CELL_PCH is not yet widely implemented. Our results are still of general nature and do not depend on the limited number of considered RRC states.

State transitions are controlled by the RNC using rules specified in the RRC protocol and timer values set by the network operator [12], [13]. A brief description is given in the following. We consider two different state transition models, depicted in Figure 2. The first model includes the Idle, FACH, and DCH states (cf. right part of the figure). Therefore, we call it 3-state model. If the UE is in the Idle state and activity is detected (i.e. a packet is sent or received), the connection transitions to DCH state. After each transmission a timer T_{DCH} is started and it is reset whenever a new packet is sent or received. If the timer expires, the connection transitions to the FACH state. Upon entering, the T_{FACH} timer is started. If a new transmission occurs, the connection again transitions to the DCH state. If T_{FACH} expires, the

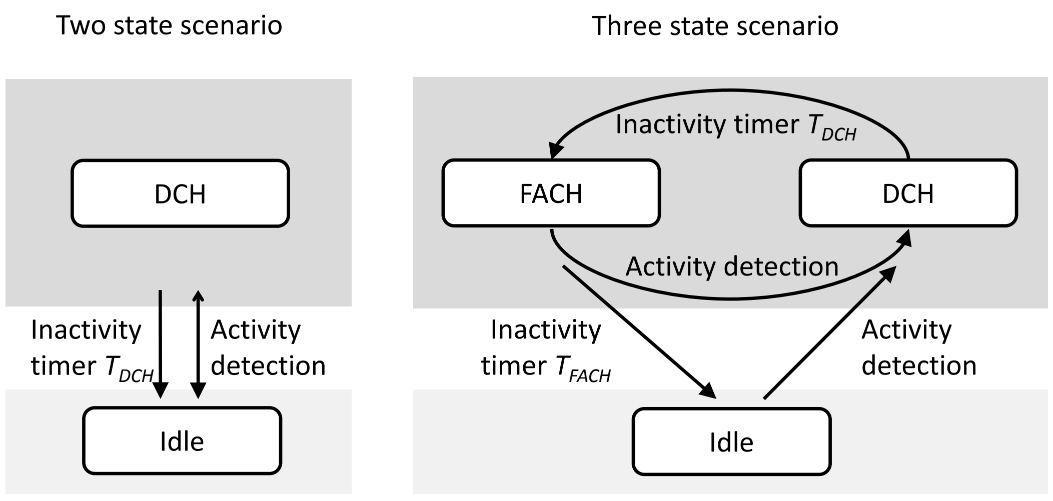


Figure 2: RRC state transition diagrams for the 2-state and 3-state models.

connection transitions to Idle state.

The second model, denoted as the 2-state model, only includes the Idle and DCH state. If the UE is in the Idle mode and a packet is sent or received, the connection transitions to the DCH state. Once in DCH mode, the T_{DCH} timer is started and it is reset whenever a new packet is sent or received. If the timer expires, the UE transitions back to Idle state.

While the 3-state model is closer to the specified RRC protocol, the 2-state model is similar to some proprietary *Fast Dormancy* implementations used by UE vendors. In these Fast Dormancy implementations, the UE tears down the connection to the network state as soon as no data is ready to be sent for a certain time, i.e., it forces the network to transition to Idle state. In contrast to the 3-state model, there is no transition to the FACH state. If a device disconnects from the network by transitioning to the Idle state, it has to be reauthenticated before another transition to the DCH state can occur. This results in additional signalling traffic and causes more load on the network [14] due to frequent re-establishments of the RRC connection. However, this decreases power consumption on the UE since the transmission unit of the UE consumes only 1-2% of the energy in Idle state compared to the DCH state. Thus, both models warrant further investigation.

2.2 Related Work on Measurements of RRC Parameters and Optimisation of Resource Consumption

The increased use of mobile broadband networks has caused radio resource management to become a hot research topic. In particular, a considerable research effort is spent on measurements of RRC configurations observed in real UMTS networks. The authors of [12] present a measurement tool exactly for that purpose. They use round-trip times of data packets to infer the RRC state of the connection. This is possible since the latency in FACH is significantly higher than in DCH state. This allows the authors to measure RRC state transition parameters such as T_{DCH} and T_{FACH}, channel setup delays and paging delays in different networks. They find that the settings vary by network and give values of 1.2 seconds for the DCH release timer and values of more than one minute for the Idle timer. To validate their method of inferring the RRC states from the round-trip times, they compare the actual energy consumption of the device with the expected energy consumption in a specific RRC.

A similar approach is applied by the authors of [13]. They reach comparable conclusions but report timer values of 5 seconds for T_{DCH} and 12 seconds for T_{FACH}. Furthermore, the authors identify sets of RRC state transitions from two network providers. We used these results to define the state models in Section 2.1.

In [15] the authors propose the tail optimisation protocol. The main idea is that the applications on the UE can accurately predict whether traffic will soon be transmitted or not. This knowledge permits to avoid the unused tail of DCH periods if no further data has to be sent. If traffic activity is expected within a short time frame, the UE stays in DCH state to avoid frequent connection re-establishments and the associated signalling load. The same authors extend this idea in [16] and propose ARO, an application resource optimiser, together with an implementation for Android 2.2. This tool also includes additional features such as batching up data or increase the update rate of application to optimise their resource consumptions.

Finally, the 3GPP has released a technical report [17] about the adverse impact of mobile data applications. This report states that frequent connection re-establishments due to small data packets caused e.g. by status updates of social network or instant messaging apps can lead to problems of increased signalling load. This highlights the importance of this topic.

2.3 Smartphone Energy Consumption and Quality of Experience

In [6] the authors performed a 4 week long study with 29 participants to identify factors influencing QoE of mobile applications. The study comprises (1) data from context sensing software, (2) user feedback using an experience sampling method several times per day and (3) weekly interviews of the participants. To determine the factors of influence, the authors analyze the frequency of specific keywords in the interviews and the only surveys. It turns out that the term *battery* has the highest frequency. According to [6] this is reasonable since the battery efficiency has a strong impact on the user perceived quality, in particular when it is nearly discharged.

As a consequence of this finding, we investigate the energy consumption of a smartphone as one of the main indicators for QoE. In addition, we show that the energy consumption of the transmission unit of a smartphone can be reduced if the state of wireless connection between the UE and the RNC is set to Idle shortly after data transmission. However, this can lead to frequent connection re-establishments since some apps send or receive small status updates very often. The signalling load produced by such issues is one of the major problem in today's UMTS networks. Hence, this is clearly a trade-off between battery efficiency on the side of the end user and the network load for the network operators.

There are different approaches to cope with such trade-offs in the design of specific mechanisms. The two most prominent ones are economic traffic management (ETM) [7] and design for tussle [8, 9]. The ETM paradigm suggests that the different stakeholders collaborate and exchange information so as to permit a joint optimisations of the trade-off. This can achieve better results than separate optimisation since these tend to work in opposite directions. Design for tussle focuses more on conflicting interests than on cooperation. It means that mechanisms should be able to adapt the outcome of a certain *tussle* at run-time and not at design-time, e.g., by giving the sole control of essential entities to a certain stakeholder.

3 Inferring Signalling Frequency and Power Consumption from Network Trace

As discussed in Section 2.1, both power consumption and generated signalling load are influenced by the RRC state transitions occurring during the devices' use. However, RRC state transitions are triggered by the UE's firmware. They can only be measured using costly hardware and on specific UEs, usually not available to researchers and developers. This prevents the application developers from evaluating the effect their applications have on the overall health on the network. Consequently, they can not take measures to prevent the harmful behaviour of their applications. However, it is possible to infer the RRC state transitions for a given packet trace if the network model is known.

In this section, we first describe the setup used to capture network packet traces for arbitrary apps. Then, we give an algorithm to infer the RRC state transitions for a given packet trace. Based on these state transitions, we can calculate the number of signalling messages generated by the packet trace. Finally, we use the information on when which RRC state was entered to calculate the power consumption of the UE's radio interface.

3.1 Measurement Procedure and Setup

To investigate the behaviour of the application under study, we capture traffic during a typical use of the application on a Samsung Galaxy SII smartphone. The smartphone runs the Android operating system and is connected to the 3G network of a major German network operator. To obtain the network packet traces we use the *tcpdump* application. This application requires *root* privileges which are obtained by rooting the device and installing the custom *cyanogenmod* ROM [18]. Once *tcpdump* is installed and running, we start the application under study and capture packet traces while the application is running. Then, the *android debugging bridge* is used to copy the traces to a workstation. The traces contain Internet Protocol (IP) packets as well as Linux Cooked Captures. We only require the IP packets, thus we filtered the traces for IP packets which are used during the following analysis.

3.2 Inferring Network State

In this section we study the influence of the application traffic on RRC state transitions and signalling messages. Since RRC state transitions can not be captures using commonly available tools, we introduce an algorithm to infer RRC state transitions from IP packet traces. Using this algorithm we analyse the RRC state transition frequency and signalling message load for the 2-state model and 3-state model.

Traffic below the network layer can not be measured without specific equipment which is often out of reach for developers interested in assessing the impact of their applications on

Algorithm 1 Inferring RRC state transitions based on IP timestamps

```
Input: Packet arrival timestamps ts
  DCH to FACH timer $T_{DCH}$
  FACH to Idle timer $T_{FACH}$
Output: Times of state transition state_time
  New states after state transitions state
  interarrival(i) $\leftarrow$ ts(i+1) - ts(i)
  index $\leftarrow$ 0
  for all ts(i) do
      if state(index) = Idle then
          index $\leftarrow$ index + 1
          state(index) $\leftarrow$ DCH
          state_time(index) $\leftarrow$ ts(i)
      end if
      if interarrival(i-1) $> T_{DCH}$ then
          index $\leftarrow$ index + 1
          state(index) $\leftarrow$ FACH
          state_time(index) $\leftarrow$ ts(i) $+T_{DCH}$
      end if
      if interarrival(i-1) $> T_{DCH} + T_{FACH}$ then
          index $\leftarrow$ index + 1
          state(index) $\leftarrow$ Idle
          state_time(index) $\leftarrow$ ts(i) $+T_{DCH} + T_{FACH}$
      end if
  end for
```

the network. Based on the 2-state and 3-state models introduced in Section 2.1 we process *tcpdump* captures of the application traffic. However, it should be noted that this method is not restricted to a specific network model, but can be extended to any other network model, as well. Using these captures, we extract the timestamps when IP packets are sent or received. Furthermore, we require the timer values of the transition from DCH state to FACH state, T_{DCH}, and the timer for the transition between FACH and Idle states, T_{FACH}. Based on these informations Algorithm 1 infers the timestamps of state transitions according to the 3GPP specification [11] for the 3-state model. This algorithm can be simplified to also work for the 2-state model. Alternatively, a way to post process the results of the algorithm to obtain results for the 2-state model is given at the end of this section. The algorithm first computes the inter-arrival times for all packets. Then, each timestamp is considered. If the UE is currently in Idle state, a state transition to DCH occurs at the moment the packet is sent or received. If the inter-arrival time exceeds the T_{DCH} timer the UE transitions to FACH T_{DCH} seconds after the packet was sent or received. Similarly, if the inter-arrival time exceeds both the T_{DCH} and T_{FACH} timers, a state transition to Idle occurs T_{FACH} seconds after the state transition to FACH.

UE vendors always search for ways to decrease energy consumption of their devices. A straightforward way to achieve this, if only the wellbeing of the UE is considered, is to transition from DCH state to Idle as soon as no additional data is ready for sending. While this transition is not directly available in the 3GPP specification for the RRC protocol [11], a UE may reset the connection, effectively transitioning from any state to Idle. This behaviour can be modelled using the 2-state model introduced in Section 2.1.

State transitions for the 2-state model can be calculate using a similar algorithm. Alternatively, the behaviour of the 2-state model can be emulated using Algorithm 1 if T_{FACH} is set to 0 and all state transitions to FACH are removed in a post processing step.

3.3 Calculating Signalling Frequency and Power Consumption

In reality, the number of state transitions is not the metric of most importance if network load should be evaluated. Each state transition results in a number of RRC messages between the UE and different network components. For this study we consider the number of messages perceived at the RNC, which can be found in [11] and is summarized in Table 1. It can be seen that transitions from or to the Idle state are especially expensive in terms of number of messages sent or received. This is due to the fact that upon entering or leaving the Idle state authentication has to be performed. Note that for the 2-state model only transitions from or to the Idle state occur. This results in the fact that for the same network packet trace the number of signalling messages occurring in the 2-state model is generally higher than in the 3-state model. To obtain the total number of signalling messages, we weight the number of state transitions with the number of messages sent per state transitions. Then, we average the number of state transitions over the measurement duration to obtain a metric for the signalling load at the RNC.

from/to	Idle	FACH	DCH
Idle	–	28	32
FACH	22	–	6
DCH	25	5	–

Table 1: Number of signalling messages per RRC state transition perceived at the RNC (taken from [11])

RRC State	Power Consumption (mW)
Idle	0
FACH	650
DCH	800

Table 2: Power consumption of the UE radio interface depending on current RRC state (taken from [16])

From a users point of view, the signalling message frequency is not important. The user is interested in a low power consumption because this increases the battery time of the device. To calculate the battery time, we use the time when state transitions occurred, and the information about the state the transition was to, to calculate the relative amount of time that was spent in each state. Given the relative time spent in each state, we use Table 2 (taken from [16]) to compute the power consumption of the radio interface during the measurement phase. To obtain the energy consumption, the power consumption can be multiplied with the duration of the measured network packet trace.

4 Numerical Results of Measurement Study

In this section we apply the methods introduced in Section 3 to four popular applications. The applications under study are introduced in Section 4.1. Then, in Section 4.2 we study how a network operator might optimise the number of signalling messages occurring at an RNC by changing inactivity timers such as T_{DCH} or T_{FACH}. In Section 4.3 we study the impact of different network parameters on the QoE perceived by the user with regard to the devices battery life. We combine these metrics and study possible trade-offs in Section 4.5. Finally, we study the influence of network parameters on web QoE metrics in terms of Mean Opinion Score (MOS) depending on page load times which is influenced by the network settings.

4.1 Characterization of Traffic Patterns for Selected Applications

For this study we chose four specific applications in order to cover a broad spectrum of traffic characteristics, as described in Table 3. First, we discuss said characteristics for these applications. We differentiate between applications, where the user interaction causes the generation of traffic, and such, where the application periodically sends or receives traffic. Finally, we consider the amount of bandwidth used by the application.

Application	Traffic characteristic	Application use	Required bandwidth
Angry Birds	Interactive	Foreground	Low bandwidth
Aupeo	Interactive	Background	High bandwidth
Twitter	Periodic, low frequency	Background	Low bandwidth
Skype	Periodic, high frequency	Background	Low bandwidth

Table 3: Characterization of applications under study

Angry Birds for Android is a popular *interactive* free-to-play game and runs in the *foreground*. To finance the game, an advertisement is shown once the player starts or restarts a level. Advertisements are downloaded on demand by the application, but require *low bandwidth*. Thus, the time between two advertisements depends on the frequency of the player advancing to the next level or deciding to restart the current one.

Aupeo is an Internet radio application, allowing a user to listen to content from personalised radio stations, while running in the *background*. Content is not streamed but downloaded at the beginning of the track. The exact duration depends on the radio stations chosen by the user and is thus *interactive*. This results in large times of inactivity during the playback of the track itself. Furthermore, due to the fact that audio files are downloaded, there is a *high bandwidth* requirement.

The Twitter client is used to send and receive new short messages from the user's Twitter account. Transferring these messages requires relatively *low bandwidth*. To this end, the user can specify an update frequency when to pull new messages in the *background*. Thus, the downloads occur with a *periodic behaviour of low frequency*. On this intervals, the client sends an HTTPS request to the Twitter server and in return receives new Tweets for the user's account. We do not consider an active user who is publishing new Tweets. Such behaviour would manifest as additional traffic to the periodic one generated by the status updates. Due to the fact that publishing updates occurs relatively infrequent, and updating the feed occurs more often, the traffic generated by publishing updates is dominated by that occurring due to updates, and thus can be neglected.

Finally, we consider the Skype application. We do not consider any Voice over IP (VoIP) calls, but the application's Idle behaviour, i.e. when the application is running in the *background*. During this time, the application sends keep-alive messages to the network. These keep-alive messages are sent with a *high frequency* and require *low bandwidth*.

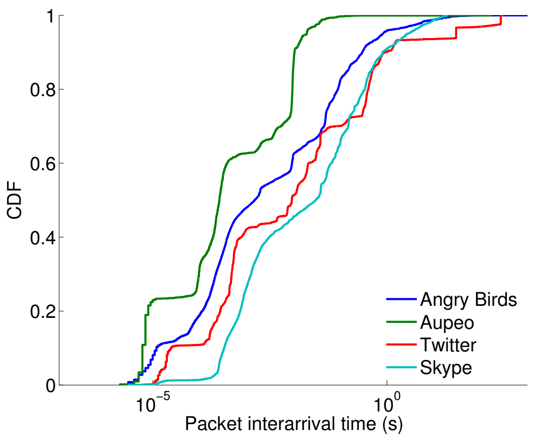
In addition to the applications considered, there exist other categories of applications which are running in the *foreground* and *interactively* require a *high bandwidth*. One example for such an application is Skype while taking a VoIP call. These applications are not considered in this study, because this kind of behaviour causes the UE to be always online. This results the minimal amount of signalling messages to be sent and a maximal power consumption at the UE, independent of network model or used parameters. Other combinations of traffic criteria also exist. However, from both a signalling load as well as a power consumption point of view, they can be mapped to one of the discussed cases. For example, if an application is sending periodic updates with low bandwidth without user interaction, then the fact that the application is running in the foreground or the background is without consequence for the generated signalling load or power consumption. However, these cases should be considered when optimisation strategies for message sending are under study. For example background applications could allow for the batching of messages, because the transmission is usually not urgent, while foreground applications do not allow for such behaviour because it would decrease QoE.

Next, we describe the applications under study in more detail. For each application we show the Cumulative Distribution Function (CDF) of the interarrival times in Figure 3a and give information about the mean values and standard deviation of both interarrival times and bandwidth in Table 4.

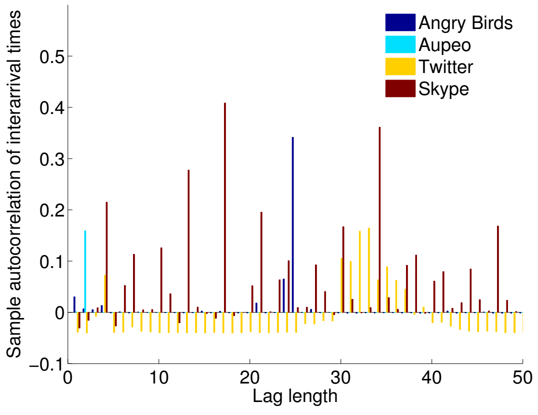
Application	Mean interarrival Time (seconds)	Standard deviation of interarrival time (seconds)	Mean bandwidth (kilobit/second)	Standard deviation of bandwidth (kilobit/second)
Angry Birds	0.66	15.90	4.42	4.50
Aupeo	0.06	3.06	129.76	482.63
Twitter	8.91	44.09	0.27	0.04
Skype	0.55	1.95	1.30	1.84

Table 4: Mean and standard deviation of interarrival time and bandwidth for considered apps

Let us again begin with the Angry Birds application. We see that there are no distinct peaks



(a) CDF of interarrival times



(b) Autocorrelation of interarrival times

Figure 3: Application characteristics

in interarrival time, which would hint at periodic behaviour. Furthermore, we see that 5% of all interarrival times are greater than 1 second. As we consider only T_{DCH} values over 1 second, those are candidates for triggering state transitions. The mean interarrival time is 0.66 seconds, with a relatively high standard deviation of 15.90 seconds. This is caused by the low interarrival times in one advertisement request and the relatively large interarrival times between two advertisements. Mean bandwidth is relatively low with 4.42 kilobit/second and a high standard deviation of 4.5 kilobit/second. These differences can be explained by considering the behaviour of the application. During long phases of use no traffic is sent, and after a level is restarted, a new advertisement has to be obtained, causing the transmission of data.

Next, we study the behaviour of the Aupeo application. We see that the application generates packets with relatively small interarrival times. This finding is backed up by the small mean interarrival time of 0.06 seconds. The high standard deviation of 3.06 seconds is caused by the wait between two tracks. Furthermore, we see a high mean bandwidth of 129.76 kilobit/second, and a standard deviation of 482.63 kilobit/seconds. This is caused by the difference in traffic activity between times when tracks are either downloaded or not.

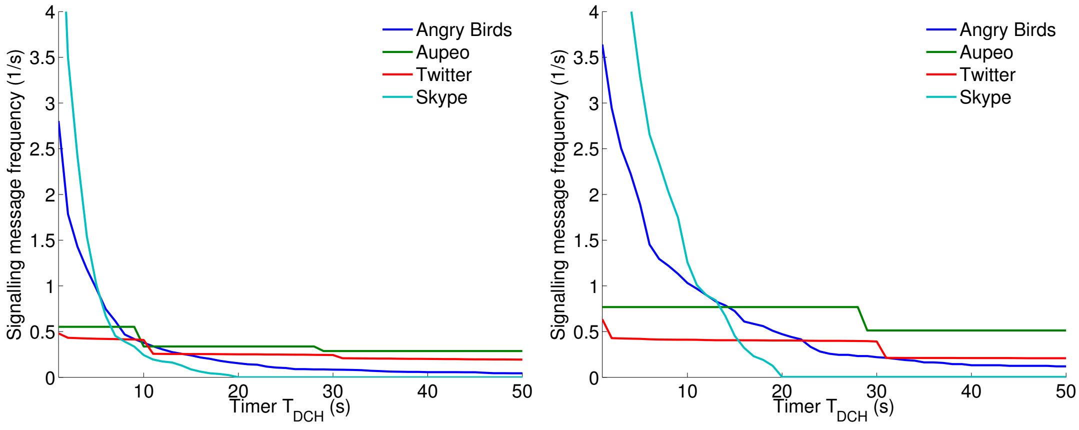
For the Twitter application we see that 90% of all transmissions occur with an interarrival time of 1 second. Also, we can observe a high mean interarrival time of 8.91 seconds and a high standard deviation of 44.49 seconds. Additionally, the mean bandwidth is low with only 0.27 kilobit/seconds and a low standard deviation of 0.04 kilobit/second due to the fact that Twitter text messages are only 140 characters in length and thus only a low volume of traffic needs to be transmitted.

Finally, we consider the Skype application. Similar to the Twitter application, we see that 90% of all packets occur with an interarrival time of less than 1 second. However, in contrast to Twitter, we see a low mean interarrival time of 0.55 seconds, with a standard deviation of 1.95 seconds. Furthermore, we observe a relatively low mean bandwidth of 1.30 kilobit/second and a standard deviation of 1.8 kilobit/second.

To further study the traffic patterns of the applications, we study the autocorrelation of the packet interarrival time with regard to the lag length in Figure 3b. We note that all studied applications present completely different autocorrelations for the interarrival times. This is one of the reasons that the applications under consideration will display different signalling behaviour in the next section.

4.2 Modifying the T_{DCH} Timer to Reduce Signalling Frequency

In light of network outages caused by so called *signalling storms* (a large number of signalling messages leading to overload at network equipment, for example an RNC) it is in the interest of a network operator to reduce the number of signalling messages arriving at the RNC. To this end, the network operator may try to modify network timer values like T_{DCH} and T_{FACH}. This results in less state transitions and thus decreases the number of signalling messages at the RNC. The result of this behaviour is shown in Figure 4. We study the frequency of signalling messages for the considered applications by applying Algorithm 1 and obtain the number of state transitions as described in Section 3.3.



(a) 3-state model (b) 2-state model

Figure 4: Signalling message frequency for varying T_{DCH} timers

4.2.1 3-State Model: Signalling Frequency

First, we study the signalling frequency generated by the studied applications if the network is using the 3-state model in Figure 4a. The figure shows the signalling frequency with regard to the T_{DCH} timer. For all studies of the 3-state model, the FACH timeout is set to $T_{FACH} = 2 \cdot T_{DCH}$, a realistic value as shown in [16]. For the Angry Birds application, we see that for the lowest considered T_{DCH} value of 1 second, in mean 2.80 messages per second are generated. This frequency drops relatively fast, for example at a T_{DCH} value of 10 seconds only 0.37 messages per second are generated. The frequency drops further until for the last considered T_{DCH} value 50 seconds, a frequency of 0.04 messages per second is reached. Next, we consider the Aupeo application. We note that even for the lowest T_{DCH} value of 1 second, we calculate a much lower signalling frequency of 0.55 messages per second. We observe a drop of signalling frequency at $T_{DCH} = 10$ seconds to 0.33 messages per second. This value decreases even further at a T_{DCH} setting of 29 seconds, where only 0.28 messages per second are generated. Then, it is maintained for all other considered values of T_{DCH}. For the Twitter application, we see that the signalling frequency starts at an even lower value for a T_{DCH} value of 1 second with 0.48 messages per second. The signalling frequency continuously drops until it reaches 0.19 messages per second for the largest considered T_{DCH} value of 50 messages per second. Last, we consider the Skype application in Idle mode. For a T_{DCH} value of 1 second, the Skype application generated 5.97 messages per second, which is not displayed on the graph. The signalling frequency drops until it reaches a value of 0.003 messages per second at a T_{DCH} value of 20 seconds. The signalling frequency shows no further significant decreases for all higher considered T_{DCH} values. This behaviour can be explained by the fact that the Skype application sends keep-alive messages with an interval of less than 20 seconds. If the timer is greater than the interval time of the keep-alive messages, the UE stays always connected and thus generates almost no signalling. These results show that the traffic patterns of the application have a large influence on the generated signalling load. Signalling is generated for every pause in sending or receiving larger than the configured timeouts. If such pauses occur frequently, this increases the signalling load as shown on the examples of Skype and Angry Birds. Applications with more time between the sending or receiving of data cause less signalling, as shown by Aupeo and Twitter. Furthermore, we can observe that the signalling load can be reduced by increasing the DCH timeout, with the minimum being reached as T_{DCH} approaches infinity. From a signalling load perspective, a value of 20 seconds would probably be sufficient, however if other metrics such as radio resource consumption are considered 10 seconds would be acceptable for a network operator.

4.2.2 2-State Model: Signalling Frequency

Next, we consider the behaviour of a network using the 2-state model in Figure 4b. The 2-state model occurs in reality if Fast Dormancy implementations are considered. Here, the UE disconnects from the network if for a certain time no traffic is sent or received in order to reduce power consumption. As for the 3-state model, the figure shows the signalling frequency with regard to the setting of the T_{DCH} timer. The Angry Birds application generates a signalling frequency of 3.64 messages per second at the smallest T_{DCH} value of 1 second, which is 125% of the signalling frequency of the application for a similar timer value at the 3-state model. This frequency continues to decrease, and at a T_{DCH} value of 10 seconds a signalling frequency of 1.03 messages per second is generated, which is 270% of the value of the 3-state model. This example shows the impact of Fast Dormancy implementations on signalling load and confirms why [14] highlights the influence of Fast Dormancy on signalling storms. Finally, at the largest considered T_{DCH} value of 50 seconds, we calculate a signalling frequency of 0.11 messages per second, which is 4 times higher than the value calculated for the 3-state model. For the Aupeo application we get a signalling frequency of 0.63 messages per second for a T_{DCH} timer value of 1 second. This is 115% of the value for the 3-state model. At a T_{DCH} value of 28 seconds the signalling frequency drops to 0.51 messages per second, 182% of the frequency of the 3-state model. This frequency is maintained for all other considered T_{DCH} values. For the Twitter application, we calculate a signalling frequency value of 0.63 messages per second at a T_{DCH} value of 1 second, which is more than 114% of the value obtained for the 3-state model. The signalling frequency decreases until 0.20 messages per second are reached for the largest considered T_{DCH} value of 50. The increase compared to the 3-state model is moderate with only 104%. For the Skype application in Idle mode we get a very high signalling frequency of 9.39 messages per second, again not shown in the figure, at a T_{DCH} timer of 1 second. This is an increase of 157% compared to the value of the 3-state model. This decreases until a T_{DCH} value of 20 seconds is reached where a signalling frequency of 0.003 messages per second is generated. Due to the behaviour of the Skype application as explained above, this results in no increase when compared to the 3-state model.

4.2.3 Lessons Learned: Signalling Frequency

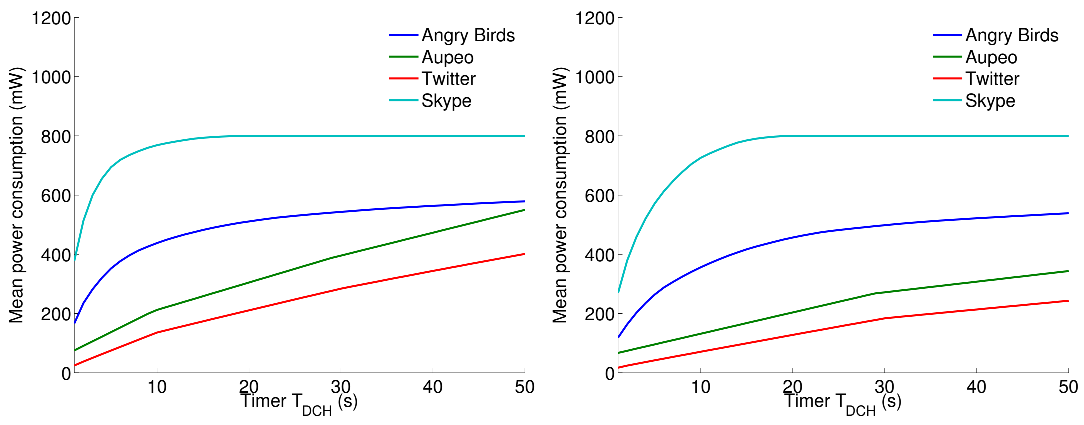
Based on this finding, we see that increasing the T_{DCH} timer decreases the signalling frequency at the RNC. However, the actual signalling frequency depends on the application running at the UE. From a network operator's point of view, the 3-state model should always be preferred to the 2-state model because it generates less signalling messages per second, thus decreasing the load at the RNC. This view does however not consider the additional radio resources which are kept in use for a longer time if larger T_{DCH} values are used. Additionally, it should be noted that the choice of the network model is sometimes outside of the domain of the network operator. Proprietary Fast Dormancy algorithms, as the considered 2-state model, are enabled on the UE by the user. To avoid the case that a user enables Fast Dormancy on his UE, the network operator needs to make sure that his network configurations do not encourage the user to enable Fast Dormancy. The impact of network timers on metrics of relevance for the user are discussed in the next section.

4.3 Energy Consumption as a Major QoE Factor

As discussed in Section 2.3, the QoE a user perceives while using his device is influenced by the battery life of the UE. In this section we study the influence of the used network model and associated timer settings on the device's power consumption. First, we consider the influence of the 3-state model on the power consumption, then we study the 2-state network model. For each of the network models, we consider the influence of network timer values T_{DCH} and, if applicable, T_{FACH} on the power consumption of the radio interface. As described in Section 3.3, based on a measurement trace for an application we use Algorithm 1 to infer the state transitions occurring during the use of the application. Then, we calculate the relative time spent in each state and use Table 2 to compute the mean power consumption of the radio interface during the measurement.

4.3.1 3-State Model: Power Consumption

In Figure 5a we consider the power consumption if the network uses the 3-state model, i.e. the Fast Dormancy mode of the UE is disabled. The figure shows the mean power consumption



(a) 3-state model

(b) 2-state model

Figure 5: Power drain for varying T_{DCH} timers

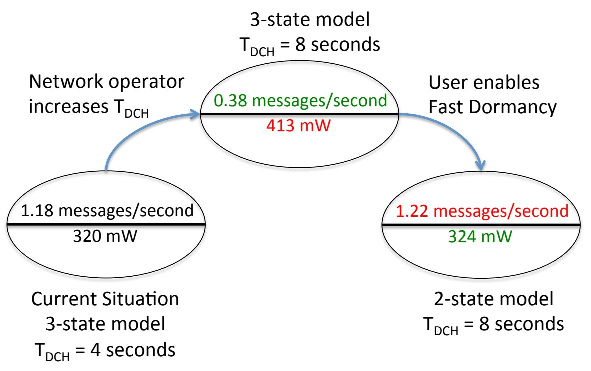
of the device with regard to the T_{DCH} timeout. Possible values range between 0 mW if the UE was in Idle state during the whole measurement and 800 mW if the UE was in DCH state during the complete measurement. For the Angry Birds application, the power consumption starts at 166.7 mW for the lowest T_{DCH} value of 1 second. It reaches a power consumption of 437 mW for a T_{DCH} value of 10 seconds. For the highest considered T_{DCH} value of 50 seconds, a power consumption of 578 mW is reached. If we consider the Aupeo application, we see that for $T_{DCH} = 1$ second 75 mW power is consumed. At $T_{DCH} = 10$ seconds, this has increased to 212.3 mW. For the largest T_{DCH} timer considered, at 50 seconds, we calculate a power consumption of 550.3 mW. The Twitter application results in the overall lowest power consumption, regardless of the chosen T_{DCH} value. For a T_{DCH} timer of 1 second we observe a power consumption of 24.4 mW. If the T_{DCH} timer is increased to 10 seconds, the power consumption increases to 135.9 mW. Finally, for the largest considered T_{DCH} value we calculate a power consumption of 401.5 mW. Finally, we consider the Skype application. At a T_{DCH} value of 1 second, the radio interface consumes 377.8 mW, the power consumption increases until at T_{DCH} a power consumption of 800 mW is reached. This is because, due to the periodic traffic behaviour of Skype, the device is always in DCH state. Again, we see that the traffic characteristics of the applications impact the power consumption. Applications with more network activity are forced to stay in more power consuming states for a longer time. We see that for very small network timers, the power consumption is minimal. However, as seen in the last section small timers increase the signalling load at the RNC. Again, a choice of 10 seconds for the T_{DCH} timer can be seen as a compromise between signalling load and power consumption.

4.3.2 2-State Model: Power Consumption

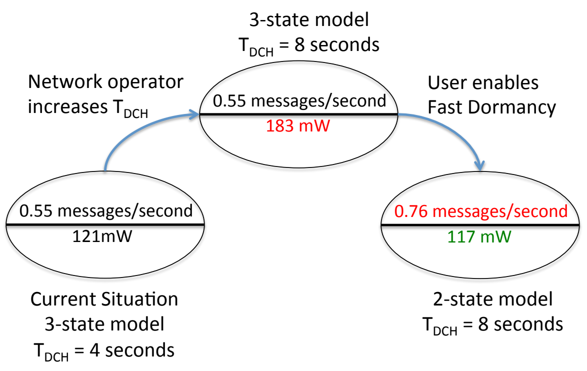
Next, we consider the changes in the power consumption of the UE if the user decides to enable Fast Dormancy, i.e. switch to a 2-state model. If we consider the Angry Birds in the 2-state network model, we get a power consumption of 118.4 mW for $T_{DCH} = 1$ second. This is a decrease to 71% of the original value. At $T_{DCH} = 10$ seconds the power consumption is 356.1 mW, which is 81% of the value of the 3-state model. At the highest considered T_{DCH} value of 538.7 mW, only 93% of the original power consumption is used. Next, we compare the results of the Aupeo application in the 2-state model with the results of the 3-state model. We see, that for a T_{DCH} value of 1 second, 67.1 mW are used, a decrease to 89% of the original value. At $T_{DCH} = 10$ seconds only 131.9 mW are consumed, which is 62% of the power consumption during the use of the 3-state model. For the largest T_{DCH} value of 50 seconds, 343.6 mW are used, which is 62% of the value for the 3-state model. If we consider the Twitter application, we again notice that the overall power consumption is the lowest irregardless of all considered T_{DCH} values. If the lowest considered value $T_{DCH} = 1$ second is chosen, we get a power consumption of 17.1 mW, an decrease to 70% of the original value. At $T_{DCH} = 10$ seconds, the power consumption has increased to 71.01 mW, which is 52.9% of the value in the 3-state model. For a T_{DCH} value of 50 seconds, the power consumption is 243.2 mW, a decrease to 60.5% of the original value. If we consider the Skype application in the 2-state mode, we see that for a timer value of $T_{DCH} = 1$ second a power consumption of 269.1 mW occurs. Again, the power consumption increases until the maximum of 800 mW is reached at a value of $T_{DCH} = 20$. Because the UE is always in DCH state as well, there is no difference when compared to the 3-state model.

4.3.3 Lessons Learned: Interaction between Mobile Operator and User

In conclusion, we see that the attempts of the network operator to optimise signalling frequency may have unexpected and undesirable effects. Consider the following simple example shown in Figure 6a. A network operator is currently using a 3-state model with a T_{DCH} value of 4 and is experiencing a signalling frequency of 1.18 messages per second and UE due to many users playing the popular Angry Birds game. To cope with this signalling load, the network operator increases the T_{DCH} threshold to 8 seconds, thus decreasing the signalling frequency per user to 0.39 messages per second, a decrease of 66.9%. However, while this is favourable for the network operator, it has negative effects for the user. While for the old setting of $T_{DCH} = 4$ seconds, the radio interface of the UE consumed 320 mW, after the increase of T_{DCH} to 8, the radio interface now consumes 413 mW, an increase of 29%. This increase may cause the user to enable the Fast Dormancy mode, which is equivalent to the network using the 2-state mode. Using the 2-state mode with a T_{DCH} timer of 8 seconds results in a power consumption of the radio interface of 324 mW, which is only marginally above the original power consumption and a decrease of 21.4% from the 3-state model. From the network operators point of view however, the use of the 2-state model results in a signalling message frequency of 1.22 messages per second, which is even worse than the original signalling frequency with a T_{DCH} value of 4 in the 3-state network. The attempt to optimise the network of the operator has caused the overall condition to worsen, for both the network operator and the end user. This seems to be a classic game theoretic optimisation problem, with the objective to find a configuration, so that neither the user has an incentive to enable the Fast Dormancy mode, nor the network operator has an interest to further increase the T_{DCH} timer. However, this optimisation problem depends on the traffic characteristics of the applications used in the network.



(a) Results of the optimisation on target application Angry Birds



(b) Influence of the optimisation for Angry Birds on Aupeo

Figure 6: Influence of manipulating T_{DCH} timer on different applications

To illustrate this, we compare the influence of the T_{DCH} timer on an application with different traffic characteristics, for example the Aupeo application as shown in Figure 6b. The signalling load before the increase of the DCH timer was 0.55 messages per second, after the change to $T_{DCH} = 8$ the load remains unchanged. Thus, the policy change based on one application brings no significant gain to other applications. However, from a users point of view, the power consumption increased from 121 mW to 183 mW. Again, we assume the user activates fast dormancy to deal with the increase in power consumption of more than 50%. This results in a decrease of power consumption to 117 mW, and an increase of overall signalling frequency to 0.76 signalling messages per second. By changing the value without considering all applications, the network operator has decreased the QoE for other users, and worsened his overall situation. Thus, due to the large number of applications it seems impossible to optimise the DCH timeout to reduce the signalling message frequency without negatively impacting the users QoE in unexpected ways.

4.4 Influence of Application Characteristics on Optimisation with Network Timers

In order to substantialize our finding from the last section that optimisation of signalling frequency using DCH timeouts based on one application may have negative impact on other applications, we study the influence of the DCH timer on both the signalling message frequency and power consumption at the same time. As in the last sections, in Figure 7a we first consider the 3-state model for DCH timers between 1 second and 50 seconds. Then, we compare these results with those obtained for a 2-state network model in Figure 7b. Calculation of both signalling message frequency and power consumption is done as described in the previous sections.

4.4.1 3-State Model: Signalling Frequency vs. Power Consumption

First, we consider the 3-state model. The X-axis of the figure gives the signalling message frequency. On the Y-axis we show the power consumption. Different T_{DCH} values are shown

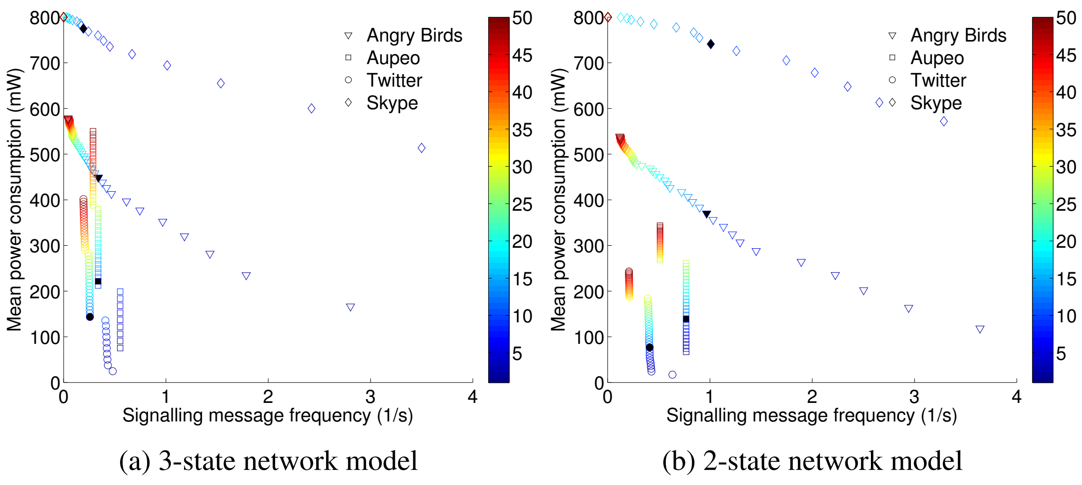


Figure 7: Influence of manipulating T_{DCH} timer on signalling message frequency and power consumption. Filled marker highlights $T_{DCH} = 11$ seconds.

by different colors as specified by the colorbar. The Angry Birds application is shown by downward pointing triangles. For increased values of T_{DCH} the signalling message decreases. However, as the signalling frequency approaches zero, the power consumption rapidly increases, even if only small gains in signalling frequency reduction can be achieved. The Aupeo application presents a completely different picture. Here, we can see multiple almost horizontal lines of markers. If T_{DCH} is chosen in this range, each increase of T_{DCH} brings a small decrease in signalling frequency for a increase in power consumption. However, some points of discontinuity exist. If for example the DCH timer is increased from 10 seconds to 11 seconds, a decrease in power consumption of 40% can be achieved by only suffering from a small increase of power consumption. These points of discontinuity would present themself to be suitable targets of optimisation. Next, we consider the Twitter application. It displays a similar behaviour as the Aupeo application, with multiple points of discontinuity. Note that Twitter exhibits a different point of discontinuity, and the T_{DCH} value of 10, which provided good results for Aupeo is not optimal for Twitter. Finally, Skype shows a completely different picture. First, note that due to the large signalling frequency of Skype for small values of T_{DCH}, $T_{DCH} = 1$ second is not displayed in the figure. Furthermore, as the T_{DCH} timer increases above 20 seconds the signalling frequency does not decrease any further, and the power consumption remains at the maximum value.

4.4.2 2-State Model: Signalling Frequency vs. Power Consumption

We compare these results with the signalling frequency and power consumption generated by the 2-state model. For the Angry Birds application we see similar behaviour as with the 3-state model, however the overall signalling frequency has increased and the overall power consumption has decreased. If we consider the Aupeo application, we can again observe the points of discontinuity. However, we note that a value of $T_{DCH} = 11$ seconds, which provided a good trade-off between power consumption and signalling frequency, is no longer optimal for the 2-state model. The Twitter application shows similar behaviour as with the 3-state model, but signalling message frequency and power consumption are increased. While in the 3-state model $T_{DCH} = 10$ seconds was still acceptable, if not optimal, in the 2-state model it is no longer acceptable. If we finally consider Skype, we note that only values above $T_{DCH} = 5$ seconds are displayed, because of the high signalling load smaller values produce. Furthermore, we see that the suggested value of $T_{DCH} = 10$ seconds generates a relatively high signalling message frequency and still consumes much energy, thus providing no good trade-off.

4.4.3 Lessons Learned: Signalling Frequency vs. Power Consumption

There exist applications, like Twitter and Aupeo, where optimisation by modifying the T_{DCH} values can provide acceptable results. However, these optimisations are only successful if a single application or network model is considered. For other applications, like Angry Birds or Skype, this optimisation approach does not seem to be successful. A reduction of signalling load and power consumption is possible, if the application developers are incentivised to optimise their applications in these regards. In [16] the authors suggest methods to achieve this optimisation, for example batch transfer of advertisements for applications like Angry Birds or decreasing the refresh rate in applications like Skype. However, at the moment application developers are neither receiving incentives to optimise applications in this way, nor do hardware vendors provide interfaces to facilitate such optimisation. Such interfaces would allow application developers to schedule their data transmissions in such a way that both signalling and battery drain would be reduced. Additionally, these interfaces would need to allow the application developer to specify if sending the transmission is urgent because the application is being actively used by the user and requires the feedback of the transmission, or if the data is being sent as a regular update while the application is running in the background and can be scheduled for later transmission as suggested by [19], [20].

To bring network operators, hardware vendors, and application developers together and allow for global optimisation of all relevant metrics, holistic approaches like Economic Traffic Management or Design for Tussle, as described in Section 2.3 are required.

4.5 Impact of Network Configuration and Background Traffic on Web QoE

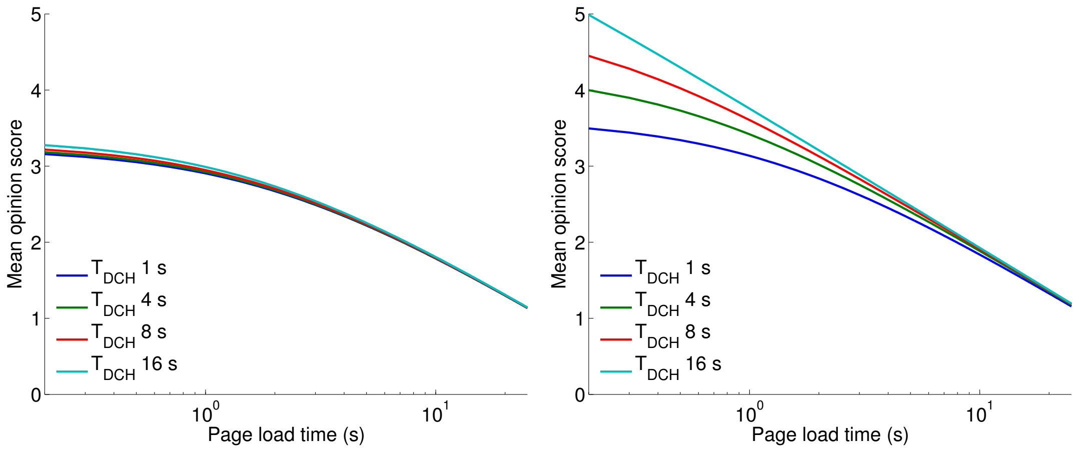
So far we have discussed only power consumption as a QoE influence factor. For applications like web browsing, one relevant QoE influence factor are page load times. Therefore, we consider a web QoE model which quantifies the impact of page load times on mean opinion scores [10]. We distinguish here between *web QoE* and *QoE*, as no QoE models are currently existing which consider page load times as well as power consumption. In this section, we study the impact of background traffic as well as network timer settings on the page load time of an image and the resulting MOS. For this study, we only consider the 3-state network model, but the results can be applied to the 2-state model as well.

We assume a scenario, where a user is running a background application like Twitter or Skype. Then, while the application is in the background, the user begins to download an image from a website. Due to the background traffic, and depending on the network model and associated timer values, the UE may be currently either in Idle, FACH or DCH state. We give the probability of a random observer encountering the system in FACH state by p_{FACH} and the probability of a random observer encountering in Idle state by p_{Idle}. If the devise is currently not in DCH state, it takes some time to connect. This promotion time depends on the current state and is according to [15] 2 seconds if the UE is in Idle state and 1.5 seconds if the device is in FACH state. For this study, we assume that the user randomly choses a time to begin downloading an image. The time until the image is displayed consists of the time to load the page t_p, as well as the time to go online t_o, where t_o is the mean time to go online, given as

$$t_o = p_{Idle} \cdot 2.5\,\mathrm{s} + p_{FACH} \cdot 1.5\,\mathrm{s}.$$

Thus, the total time t that is required to download the image is given by $t = t_o + t_p$.

The authors of [10] give a function to calculate the MOS based on the required page load time as $QoE(t) = a \cdot \ln t + b$, were a and b depend on the type of content being downloaded. For our scenario, picture download, values of $a = -0.8$ and $b = 3.77$ are suggested. It has to be noted that for different web sites, the logarithmic function was still observed, but different values for a and b were obtained as given in [10]. These values depend for example on the type of web page as well as the size of the content. Nevertheless, the results presented in this section are therefore generalizable for web browsing to various pages. This allows us to give an expected MOS for downloading pictures while a background application is influencing the probability of a device already being in DCH state or still having to be promoted to DCH state.



(a) Background traffic generated by Twitter (b) Background traffic generated by Skype

Figure 8: Perceived QoE for loading a page with existing background traffic

Using this methodology we study the influence of background traffic on the QoE for two background applications with different traffic characteristics. In Figure 8a we assume that the user is running the Twitter application as a background process. The application is set to update the users status feed every 5 minutes. In Figure 8b the user is running the Skype application as a background application. This application sends keep alive messages every

20 seconds. For each application, we assume the 3-state network model with T_{DCH} settings of $1, 4, 8$, and 16 seconds. We always set $T_{FACH} = 2 \cdot T_{DCH}$. In both figures we show the assumed page load time, as provided by the network, on the X-axis for values from 0.2 to 25 seconds. We assume 0.1 seconds as a lower bound because page load times lower than 0.1 seconds are not distinguishable [21] by humans. The calculated MOS values are given on the Y-Axis.

The picture downloads with the background traffic generated by the Twitter application result in MOS values beginning at 3.15 for $T_{DCH} = 1$ second, 3.18 for $T_{DCH} = 4$ seconds, 3.21 for $T_{DCH} = 8$ seconds and 3.27 for $T_{DCH} = 16$ seconds. With increasing page load time, the MOS again decreases. This behaviour is due to the fact that the Twitter application periodically sends traffic every 5 minutes. Then, no further activity occurs until the next refresh occurs. In this time, the UE transitions to Idle state. This traffic characteristic causes a high probability of a user encountering the device in a Idle state. In contrast, downloading pictures with the Skype application generating background traffic, causes different MOS values. For a page load time of 0.2 seconds the MOS value with $T_{DCH} = 1$ seconds is 3.49, with $T_{DCH} = 4$ seconds we get 3.99, for $T_{DCH} = 8$ seconds we get a MOS value of 4.44, and finally for $T_{DCH} = 16$ seconds we get 4.99. For increasing page load times, the MOS decreases. This increased MOS values occur because of the high frequency of traffic sent by the Skype application. Here, every 20 seconds traffic is sent. This means, that even for relatively low values of T_{DCH} the user has a high probability of encountering a state where no promotion delay is required before the actual page load time can begin.

From these studies we can conclude that, when considering QoE on mobile devices, not only the page load time caused by the network but also additional delays caused by the state of the device should be considered. As shown on two examples, this state can be affected by other applications which are running in the background and generate traffic.

5 Conclusion

In this study we investigate the influence of both application traffic characteristics and network configuration on the signalling load at the RNC as well as the QoE for the user. We consider two different influence factors on QoE: power consumption and page load times for web browsing.

First, we consider the power consumption at the UE. The network operator can reduce the signalling load at the RNC by increasing a network parameter, the T_{DCH} timer, which determines the time a UE remains connected to the network. However, increasing this timer also increases the battery drain at the UE. While it is possible to find a trade-off between signalling load and battery drain for any single application, we show that for a set of applications, this is not possible. A network parameter, which might be optimal for one application, may cause another application to generate either a high signalling load or an increased battery drain. Thus, optimisation using this parameter is not successful, and alternative means need to established. First, application developers could consider the effect of the traffic their applications produce both on the UE and the network. To this end, we will provide a web application as future work, which will allow application developers to upload and analyse their applications' traffic according to different network models. Secondly, the UE vendor could provide interfaces for application developers to use when sending data. These interfaces would schedule data to be transmitted in such a way that signalling load and battery drain would be reduced, if the application's requirements allow for it. Finally, approaches like *Economic Traffic Management* or *Design for Tussle* could be applied to find an acceptable trade-off for all parties. In Economic Traffic Management all participating entities share information in order to enable collaboration. This collaboration allows for a joint optimisation of the trade-off. Design for Tussle aims to resolve tussels at run time, instead instead of design time. This prevents the case that one actor has full control over the optimisation problem, which would likely result in the actor choosing a trade off only in its favour, ignoring all other participants.

Additionally, we consider the influence of background applications on the web-QoE. For mobile applications, the page load time may be increased if the UE is currently disconnected and has to connect to the network before obtaining the requested content. If another application is already running, the device may, depending on the traffic pattern of the application, already be online. We suggest to consider this when performing QoE studies for mobile users.

References

[1] Google, "Eric schmidt at mobile world congress 2011," 2011.

[2] T. Muller and J. Bowcock, "Apple's app store downloads top 10 billion," Jan. 2011.

[3] Cisco, "Cisco visual networking index: Global mobile data traffic forecast update, 2011-2016," February 2012.

[4] Penn-Olson, "Finding a connection: Android, line, and docomo's network outage," 2012.

[5] The Economic Times, "Docomo to ask for changes in android os: Report," 2012.

[6] S. Ickin, K. Wac, M. Fiedler, L. Janowski, J. Hong, and A. Dey, "Factors influencing quality of experience of commonly used mobile applications," *Communications Magazine, IEEE*, vol. 50, no. 4, pp. 48–56, 2012.

[7] T. Hoßfeld, D. Hausheer, F. Hecht, F. Lehrieder, S. Oechsner, I. Papafili, P. Racz, S. Soursos, D. Staehle, G. D. Stamoulis, P. Tran-Gia, and B. Stiller, "An Economic Traffic Management Approach to Enable the TripleWin for Users, ISPs, and Overlay Providers," in *FIA Prague Book, ISBN 978-1-60750-007-0* (G. Tselentis, J. Domingue, A. Galis, A. Gavras, D. Hausheer, S. Krco, V. Lotz, and T. Zahariadis, eds.), p. 24, Towards the Future Internet – A European Research Perspective: IOS Press Books Online, May 2009.

[8] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden, "Tussle in cyberspace: defining tomorrow's internet," *IEEE/ACM Trans. Netw.*, vol. 13, pp. 462–475, June 2005.

[9] "D2 - Lessons in 'Designing for Tussle' from Case Studies." Deliverable of the Trilogy Project ICT-216372, 2008.

[10] S. Egger, P. Reichl, T. Hoßfeld, and R. Schatz, "Time is Bandwidth? Narrowing the Gap between Subjective Time Perception and Quality of Experience," in *2012 IEEE International Conference on Communications (ICC 2012)*, (Ottawa, Canada), June 2012.

[11] "Ts 25.331, radio resource control (RRC); protocol specification," 2012.

[12] P. Perala, A. Barbuzzi, G. Boggia, and K. Pentikousis, "Theory and practice of RRC state transitions in UMTS networks," in *Proceedings of the global communication conference (GLOBECOM) Workshops*, 2009.

[13] F. Qian, Z. Wang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck, "Characterizing radio resource allocation for 3g networks," in *Proceedings of the 10th annual conference on Internet measurement (IMC)*, 2010.

[14] NokiaSiemensNetworks, "White paper: Understanding smartphone behavior in the network," 2011.

[15] F. Qian, Z. Wang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck, "Top: Tail optimization protocol for cellular radio resource allocation," in *Proceedings of the 18th IEEE international conference on network protocols (ICNP)*, 2010.

[16] F. Qian, Z. Wang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck, "Profiling resource usage for mobile applications: a cross-layer approach," in *Proceedings of the 9th international conference on mobile systems, applications, and services (Mobisys)*, 2011.

[17] "Tr 22.801, study on non-mtc mobile data applications impacts (release 11)," 2011.

[18] "Cyanogenmod," 2012.

[19] M. Calder and M. Marina, "Batch scheduling of recurrent applications for energy savings on mobile phones," in *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on*, pp. 1–3, IEEE, 2010.

[20] E. Vergara and S. Nadjm-Tehrani, "Energy-aware cross-layer burst buffering for wireless communication," in *Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet*, p. 24, ACM, 2012.

[21] S. Egger, T. Hoßfeld, R. Schatz, and M. Fiedler, "Waiting Times in Quality of Experience for Web Based Services," in *QoMEX 2012*, (Yarra Valley, Australia), July 2012.