Energy-efficient internetworking with DTN

Dimitris Vardalis and Vassilis Tsaoussidis

Space Internetworking Center, Dept Of Electrical and Computer Engineering,

Demokritos University of Thrace, Xanthi 67100, Greece

Abstract. We claim that DTN has the potential to form an internetworking overlay that shapes traffic in a manner that exploits the capacity of last hop wireless channels and allows for energy-efficient internetworking. We demonstrate DTNs potential for energy-efficient internetworking through an overlay-architecture and a tool we have developed that captures the required state transitions of the mobile device WNIC. We show experimentally that the DTN overlay can re-shape traffic in a manner that allows the receiver to exploit the energy-throughput tradeoff better: it may condense sporadic packets into a burst and, in return, prolong sleep and power-off duration without risk to miss incoming packets and without degradation in throughput.

Keywords: DTN, energy efficiency, internetworking

1. Introduction

In recent years, the need for mitigating energy expenditure has been even more acute. Modern mobile devices are equipped with increasingly more powerful CPUs, larger memory capacities, faster wireless network adapters, and a set of applications that require internetworking capabilities. Hence, mobile devices became a useful every-day tool for communication, storage and entertainment, which, in turn, increased rapidly their frequency of use, from the order of minutes a few years ago, to several hours per day, recently. However, advances in battery technology have not been able to match the increased energy requirements: users had to adjust their behavior according to the device capacity and application developers faced a bottleneck, since application sophistication is confined by the operational capacity of the devices.

So far, the main focus of attention was the energy-efficient administration of device components, the selection of routing with energy capacity to route packets, or the capacity of battery itself. The (inter-) network *per se* has never attested its vital role in energy efficiency. We claim here that the Internet itself, coupled with a supportive Delay Tolerant Networking (DTN) overlay [1], [2], could become a tool for shaping traffic destined to mobile devices in an energy-efficient manner.

Presently, the Internet is the major reason for both the lack of end-device energy control (due to its probabilistic communication nature) and for extra energy expenditure (due to the extended communication time required by the internetworking applications). Typically, although transmission scheduling and, hence, device operational modes can be largely predicted in a deterministic environment, where communication slots can be adjusted to a given contact graph, internetworking is dominated by uncertainty in scheduling packet transmission and reception. This uncertainty forces a receiver on continuous waiting for random contacts and, inevitably, enforces an energy-wasting policy; as a result, the device cannot incorporate energy-efficient modes of operation, such as sleep, into its transition state diagram. Therefore, a main reason for the inability to optimize end-device operations is the inherent inability of the internetwork to deliver data tactically, due to its store-and-forward and statistical multiplexing characteristics. Practically, what imposes the limitation to schedule transmission with accurate precision is the variation of delay, along with the potential lack of sufficient resources to store packets in transit, albeit having such a capacity could have, instead, permitted the network to shape traffic and corroborate a scheduling policy among the devices *and* the network.

In this work, we attempt to demonstrate the potential of DTN to shape internetwork traffic in a manner that allows mobile devices to balance their energy expenditure with minimal cost on throughput. In this context, we design and deploy an internetworking overlay that exploits two major DTN properties: (i) to store packets for as long as it is necessary, regardless of disruptions in communications; and (ii) to enhance the edge nodes with functionality to wait for sufficient amount of packets to arrive, prior to transmitting to the end node. We then develop scenarios and weigh our expectations using a broad set of experiments that reflect potential variations in network delay, traffic characteristics, error probabilities and disruptions.

In order to evaluate the proposed scheme we use the ns-2 network simulator. Since there is no established and reliable DTN implementation for ns-2 we opted for emulating the behavior of the bundle protocol by introducing a proxy application at the Base Station (BS). The energy expenditure model of ns-2 was also not sufficient for this work, since the whole state transition diagram for the WNIC had to be recorded. Hence, energy expenditure calculations involved hacking into the physical layer of the wireless node and adding tracing for the state transitions. Our results show that (i) energy expenditure can be reduced drastically and (ii) clearly, DTN appears as an appropriate tool to construct such overlay: it allows for permanent storage and, in addition, it does not require full, but instead, partial deployment on top of IP; in fact, even a minimal deployment of DTN nodes can satisfy the architectural requirements of energy-saving overlay as soon as the edge nodes deploy such service.

The rest of the paper is organized as follows: In section 2 we discuss related work, focusing on more recent approaches to save energy or tackle energy using DTN. In section 3 we discuss our proposal and in section 4 we detail our experimental tool and methodology, including parameters for evaluation, metrics and scenarios developed. In section 5 we discuss the results and, finally, in section 6 we summarize our conclusions.

2. Related Work

The research literature on energy efficiency for mobile devices includes two broad categories that correspond to the device optimization and communication optimization. In [3], the authors identify as the main sources of energy expenditure the various subsystems of a handheld device, such as: the processor, the memory, the display, the audio system and the wireless networking. For network intensive applications, the power consumption of the Wireless Network Interface Controller (WNIC) can reach up to 60% of the total power necessary for the operation of the mobile device [4]. In [5] Jones et al. provide a comprehensive survey on the design principles of efficient network protocols. The authors focus on the specific mechanisms that can be employed in each of the layers in the network protocol stack (i.e. Physical, Data Link, Network, Transport, OS/Middleware and Application).

In [6], Adams et al. propose a buffering technique that exploits the inherent power-saving mode of 802.11. Data destined to a certain mobile node is hidden and thus, not made available to it for multiple beacon intervals. Balasubramanian et al. [7] perform detailed measurements on the energy expenditure of mobile devices when using 3G, GSM and WiFi as their network interface. They also introduce a protocol that delays transmissions (for delay tolerant applications, such as email) and prefetches data (for applications like web search). Mamatas and Tsaoussidis in [8] monitor the state transition of the WNIC and introduce two metrics for energy potential and unexploited available resource, to capture the energy performance potential of various protocols. Acquaviva et al. [9] explore the energy expenditure impact of a close-loop strategy (the client triggers the transmission of data) and two open-loop strategies (the server has full control of the data transmission), when streaming from a single server to multiple clients. The authors in [10] present a theoretical and empirical study of the impact of packet transmission interval and burst length on observed network performance characteristics, with or without use of a proxy service. They propose an adaptive, energy-saving streaming mechanism that adjusts the burst length. In [11] the authors show that for multimedia streaming of even moderately high bandwidth content the WNIC completely switches out of the power saving modes, since the idle intervals between received frames are not of adequate duration. They propose using an application-specific server-side proxy service for spacing out traffic bursts and they show that the energy expenditure per KB of received data can be reduced by as much as 83%. By the same token, however now focusing generally on TCP traffic, Anastasi et al. [12] introduce at the Base Station a proxy service that employs regular TCP on the wired network side and the novel Power-Saving Transmission Protocol on the wireless side. Authors in [13] expand on the proxy idea by introducing a scheduler service at the Base Station and a proxy at the mobile terminal. The scheduler incorporates a novel algorithm, called priority-based bulk scheduling (PBS). The same authors in [14] propose a similar architecture with a *rate-based bulk scheduling algorithm (RBS).* The client proxy is responsible for pre-fetching data and managing the state of operation for the WNIC. In [15] the proxy idea is again implemented at the Base Station introducing the novel RT_PS protocol for the communication between the Base Station and the mobile node. The RT_PS protocol includes idle intervals information so that the client can put the WNIC into sleep mode when applicable. By the same token, Batsiolas and

Nikolaidis [16] propose a regulation of ACKs for TCP in a manner that allows the sender to predict with some accuracy when ACKs are expected; and meanwhile the sender enters a sleep mode.

Delay/Disruption Tolerant Networking (DTN) that is employed in this work, allows for wide-spread store-and-forward strategies, beyond the Base Station. That is, the disruptive network (deployed as an overlay, here) can schedule transmissions internally aiming at maximizing the performance of the link between the edge and the receiver; flow control for the edge is not a matter of the sender alone but a strategy of the network itself. This is not a rather insignificant difference. Instead, it allows for more accurate transmission control: the Base Station is not controlled by a single application and hence each application separately cannot accurately control the BS flow, and in turn, regulate its buffering capacity. Unlike typical end-to-end services, DTN inherently employs a load balancing strategy that allows for naturally distributing network buffering.

3. Energy-efficient internetworking overlay

We depart from the mobile device *modus operandi*: at any given moment, a WNIC can be in one of five operation states, each with different power requirements: transmit, receive, idle, sleep (or doze) and power-off. The *transmit*, *receive* and *idle* state of the WNIC constitute the active states, whereas the *sleep* and *power-off* constitute the inactive states. The power requirements of active states are comparable with each other [3]. In the transmit state maximum power is consumed, while in the receive state slightly more power is required than in the idle state. In the sleep state the energy expenditure is an order of magnitude less than that of the idle state, and finally, in the power-off state no power is required.

The IEEE 802.11 [17] protocol provides an inherent mechanism for allowing client devices to manage the power requirements of their WNICs. The mechanism relies on the central role played by the Access Point when the network is in the PCF (Point Coordination Function) operation mode (in this paper we do not consider the ad-hoc or Distributed Coordination Function of 802.11). Before entering Power Saving Mode (PSM) the WNIC notifies the Access Point and then falls into the sleep state. While in this state, packets arriving at the Access Point and destined to the sleeping node are temporarily buffered, waiting for the node to wake up. The Access Point broadcasts periodical “beacon” signals that are followed by a list of node addresses, called Traffic Indication Map (TIM), for which there is buffered data available. When the sleeping node wakes up, it listens for the next Beacon and checks if its address is included in the TIM. If no data is available the node can go back into the sleep state, otherwise it transmits a PS-Poll frame and the AP responds by sending the queued data.

It becomes apparent that the challenge in energy-saving strategies now translates into how to maximize the duration that the WNIC spends in either the sleep or the power-off state. It is obvious, however, that the AP has a limited buffer and therefore, the scheduling precision and flow control of the AP determine the efficiency of each strategy. Otherwise, the risk of entering a sleep mode may lead to costly retransmissions and throughput degradation due to buffer overflow or receiver inactivity. Data buffering essentially shapes traffic in bursts so that periods of inactivity are prolonged, allowing the WNIC to stay in a low-power state longer, while at the same time minimizing the performance trade-off with throughput. Hence, it makes sense to design an overlay that buffers at least a Delay X Bandwidth product between the edge node and the receiver. In addition, a signaling or probing protocol will be necessary for the edge node and the receiver. The approach could develop more sophisticated techniques that avoid signaling: For example, a time slot for initiating transmission for the edge could become the transmission delay of a full DxB product of the last hop itself; something that can be measured during the first window of opportunity for successful transmission and poses a hybrid Time Division and Statistical Multiplexing strategy. Furthermore, the target eventually becomes to balance the network so the edge node will always have a DxB product to submit when opportunity exists but not overload it with more data than it can handle per slot, in order to avoid dropping. This sort of sophistication has not been developed or analyzed in our present work. We construct a rather demonstrative overlay where buffers have enough space; the disruptions never go beyond the storage capacity, so the possibility to cause buffer overflow is cancelled, in practice. In this context, we evaluate only the ability of the overlay to shape traffic in a manner that optimizes the transitions of the mobile device; and we evaluate the characteristics of the tradeoff with throughput. However, we note here that disruptions do occur and therefore the transmission strategy is not a pure product of time required to fill in the DxB product but conflicts with the connectivity schedule, as well.

What we consider as a productive part of our proposal is the ability of the edge node (i.e. the last DTN node) to make a decision locally, with better precision than the sender of a typical end-to-end application. The architecture has the potential for accepting easy enhancements in case it exploits further the custody property of the DTN protocol, which leads to an automated balancing of storage as a result of the capability to shift data towards the destination only when the accepting nodes do have storage availability.

```
ReceiveIncomingData
If IsBaseStationNode
{
        If receivedData Is lastSetOfData OR receivedData > bufferingThreshold
                If (HasConnectivityToMobileEndNode)FlushData
                Else
                {
                        If (BufferAvailable) StoreData
                        Else RejectDataCustody
                }
}
Else
{
                If (HasConnectivityToNextHop) ForwardData
                Else
                {
                        If (BufferAvailable) StoreData
                        Else RejectDataCustody
                }
}
Go Back to the Beginning
```

It is obvious, however, that the AP has a limited buffer and therefore, the scheduling precision and flow control of the AP determine the efficiency of the strategy. Otherwise, the risk of entering a sleep mode may lead to costly retransmissions and throughput degradation due to buffer overflow or receiver inactivity.

4. Experimental methodology

We emulated the behavior of the bundle protocol by introducing a proxy application at the base station. Measuring the energy expenditure was made possible by directly monitoring the physical layer of the receiver and making log entries for each state change of the WNIC. The proxy application is associated with an incoming TCP connection originating from some source on the wired part of the network and an outgoing TCP connection that ends at the wireless receiver. The proxy application buffers data coming in from its receiving end and flushes it out to its sending end as soon as the desired amount of data is collected. This setup emulates the functionality of a DTN overlay on an IP network, when using TCP as the convergence layer.

The proxy application was implemented as a subclass of the Application class in ns-2, adding the required extra functionality. Additional functionality implemented within the new class includes: an agent that receives data from the wired source, an option for the buffer size and an option for the total size of the transfer. The proxy needs to be aware of the total transfer size so that, when the specified amount of data is received, the buffer is immediately flushed without waiting for the buffering threshold to be reached.

In order to facilitate the energy expenditure calculations we modified the physical layer of the wireless node. Code that traces the duration the WNIC spends in each of the three states (i.e. *send*, *receive* and *idle*) was added. By having an accurate picture of state transitions we are able to carry out post simulation calculations and find the potential energy savings if there was in place a mechanism that would switch the WNIC into the *sleep* state during *idle* intervals. In this paper we aim at emphasizing the energy saving potential that a DTN-based protocol can have for mobile devices. Addressing various optimization aspects, including the inherent load balancing capability of DTN custody transfer, is part of our future work plans.

4.1 Energy expenditure calculations and comparison.

During the post-simulation analysis a multi-purpose script identifies the energy-related log entries in the ns-2 trace file and calculates the energy expenditure based on certain input parameters. The script takes the following input parameters: transmit power (`txPower`), receive power (`rxPower`), idle power (`idlePower`), sleep power (`sleepPower`), transition power (`transPower`) and transition time (`transTime`). For the *transmit* and *receive* intervals the energy expenditure is calculated as the duration of the interval multiplied by the transmit and receive power respectively. Energy conservation can be achieved by exploiting the *idle* intervals and switching into the *sleep* state. The transition is considered feasible only if the time it takes to fall into the *sleep* state and then back to the *idle* state (i.e. `transTime`) is less than the duration of the *idle* interval itself. The transition is considered meaningful if the energy required for switching back and forth to the *sleep* state, plus the energy spent during the *sleep* state is less than the energy expenditure if the WNIC had remained in an *idle* state for the whole interval duration. If `idleDuration` is the duration of the *idle* interval, the decision is based on the following formula:

```
2 * transPower * transTime + (idleDuration - 2 * transTime) *
        sleepPower < idleDuration * idlePower
```

It becomes obvious that the longer the duration of the *idle* intervals the greatest the opportunity for conserving energy. The power figures for the various WNIC states are set as follows [18]: `txPower` = 1.400 Watts, `rxPower` = 0.950 Watts, `idlePower` = 0.805 Watts and `sleepPower` = 0.060 Watts. As reported in [19] transition time from *sleep* to *idle* and vice versa, is in the order of tens of milliseconds. However, when switching back to the active state, the controller needs to wait for the next beacon in order to start receiving the buffered data. If we assume that it takes 50ms to switch states and another 100ms for the next beacon signal to be broadcast (a rather pessimistic estimation since 100ms is the default beacon interval for 802.11) the total for the transition amounts to 200ms. Therefore, we set the transition delay at 100 ms for both state transitions. Finally, based on measurements in [20] we can safely consider the energy expenditure in the transitive state to be the same as that of the *idle* state (i.e. `transPower` = 0.805).

4.2 Model description

Figure 1 depicts the network topology that was implemented in order to evaluate the energy conservation that could be achieved by employing a DTN-like protocol in wired-cum-wireless scenarios. Network nodes are named as N1 – N6. N4 is the base station node that is connected to both the wired and the wireless networks and node N6 is the wireless receiver. Links L13, L23, L34 and L45 are wired, while WL is the wireless link between the base station and the receiver.

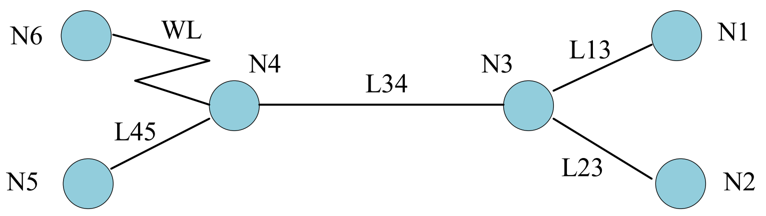


Figure 1: Network Topology.

The data transfer studied in this experiment follows the N1→N3→N4→N6 route, while the competing flow follows the N2→N3→N4→N5 route. The bandwidth of the backbone link (L34) is changed so that the desired congestion is introduced into the network. The rest of the links are kept at the same delay and bandwidth values for all the experiments.

The bandwidth and delay values for all the links are as follows: L13 - 2Mb, 100ms, L23 - 3Mb, 100ms, L34 – 300ms delay and varying bandwidth depending on the desired network characteristic, L45 - 3Mb, 100ms, WL - 802.11 with a data rate of 11Mb and a basic rate of 1Mb. Stochastic Fair Queuing is applied on all queues, the file transfer size is 10MB and, when present, the competing flow sends a 1000 byte packet every 3 ms. TCP packet size is 1460 and maximum window size 100 packets.

The resulting energy expenditure values are in Joule. In the End-to-end (E2E) scenarios there is an end-to-end connection between N1 and N6, whereas in the Split scenarios there is a splitting application on the base station acting as a transport layer mediator. In order to measure the delay of the wireless connection between the base station and the mobile receive we employed a ping agent. The delay was found to be around 2ms. The DxB is thus: 0.002 x 11 000 000 / 8 = 2.750 KB. Except otherwise noted, the splitting application buffers 30KB (around 10 x the DxB) before forwarding them to the mobile device. In case the wireless connection is the bottleneck and data piles up at the base station we make the assumption that there is no limit in the available memory used for buffering data and, therefore, no data is dropped.

Finally, experiments are carried out both for an FTP connection transferring 10 MB of data as well as CBR traffic that flows for a specified amount of time. In certain experiments we are introducing a packet error rate on the wireless LAN. The error follows a uniform distribution and is applied on the outgoing interface of both the base station interface as well as the WNIC of the mobile node.

5. Results

In this section we present the experimental results, organized in five subsections of experimental setups. The setups are as follows: FTP and varying wired delay (with and without a competing flow), FTP and varying wired bandwidth (with and without a competing flow), FTP and varying wireless error (with and without a competing flow), FTP and varying buffering threshold (with and without a competing flow), CBR and varying buffering threshold.

5.1 Varying Delay FTP Transfer

In this scenario the bandwidth of the backbone link is 2Mb (the same as the upstream link L13), there is no competing data flow and a 10MB file is transferred over FTP from N1 to N6. By assigning different delay values to the backbone link we study the impact of varying delay (i.e. on the wired part of the network) to both the duration of the transfer as well as the total energy spent.

L34 delay (ms)	Duration E2E (sec)	Duration Split (sec)	Energy E2E (Joule)	Energy Split (Joule)	Energy E2E No PSM (Joule)
100	71.70	72.52	56.21	52.03	62.97
200	68.38	68.08	51.80	49.12	60.30
300	90.48	90.18	59.05	54.73	78.09
400	112.58	112.23	61.32	56.49	95.89
500	134.68	134.38	62.19	58.31	113.68

700	178.88	178.57	66.89	61.92	149.26
1000	245.19	244.88	72.25	67.04	202.63

Table 1: FTP, Varying Delay, No Competing Flow Measurements.

It can be seen from Table 1 that the transfer durations for the E2E and the Split cases are almost identical for all delay values. Thus, a DTN-based strategy for buffering data at the BS does not necessarily mean that there will be a tradeoff in the performance, as long as there is a mechanism for detecting the end of the transfer. For values greater than 200 ms the transfer duration for all cases increases by approximately 20 seconds for every 100 ms increase of the backbone link delay.

The chart in Figure 2 shows that the energy expenditure of the Split case is consistently 3-5 Joule less than that of the E2E case, only slightly increasing with the transfer duration. Therefore, when no congestion or other type of error is present, the energy conservation gains of the Split method only marginally depend on the overall transfer duration. Transmitting the same amount of data (i.e. physical layer data) in longer intervals adds to the overall time the WNIC of the receiver spends in the Sleep state. Since the power of the Sleep state is at least an order of magnitude less than the power of the Active state, the additional energy saving for longer transfer durations is negligible. On the contrary, when no PSM is employed the energy expenditure consistently increases as the transfer duration is prolonged.

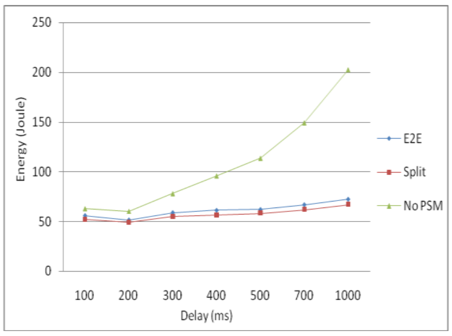


Figure 2: FTP, Varying Delay, No Competing Flow Energy Expenditure.

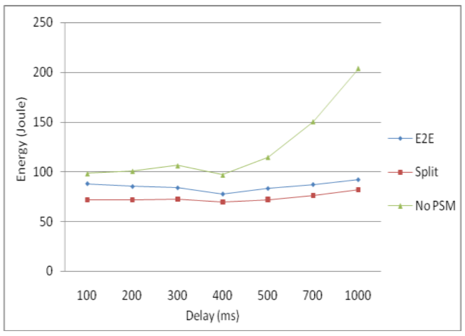


Figure 3: FTP, Varying Delay, Competing Flow, Energy Expenditure.

Running the experiments with the same parameters, but adding a competing flow that shares the backbone link with the flow under examination yields similar results. The amount of energy conserved when employing the Split approach is greater than in the no competing flow case, ranging from 8-17 Joule. However, there does not seem to be a connection between the energy savings and the transfer duration. Also, it is evident that the competing flow adds some randomness to the outcome, possibly due to different packets being dropped depending on space availability in the queue of the bottleneck link. The outcome of this set of runs is depicted in Figure 3.

5.2 Varying Bandwidth FTP Transfer

In this set of experiments the backbone link is assigned a constant delay of 300ms, while the bandwidth varies from 0.5Mb to 2.5Mb in steps of 0.5Mb. The total transfer duration (Table 2) is virtually the same for both the E2E and the Split case, and it is greatly affected by the bandwidth of the backbone link because of network congestion. Since the upstream link (L13) has a 2Mb bandwidth, setting the backbone bandwidth to values greater than 2Mb does not affect simulation results (L13 becomes the bottleneck of the connection and the backbone is uderutilized). Because of this, the transfer duration is equal for bandwidth values of 2Mb and 2.5Mb.

L34 bandwidth (Mb)	Duration E2E (sec)	Duration Split (sec)
0.5	216.85	216.05
1	145.59	144.88
1.5	111.65	111.17
2	90.48	90.18
2.5	90.34	90.04

Table 2: FTP, Varying Bandwidth, No Competing Flow Transfer Duration.

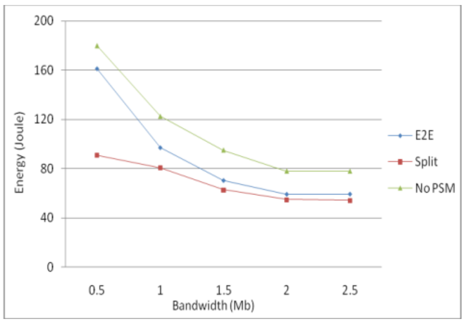


Figure 4: FTP, Varying Bandwidth, No Competing Flow Energy Expenditure.

In terms of the energy efficiency, it is obvious from the chart that smaller bandwidth values for the bottleneck link lead to greater energy gains of the Split approach. In the extreme setting of the 0.5Mb bandwidth, the energy conservation achieved in the Split case is approximately 44% of the energy expenditure in the E2E case.

As the network congestion is eased, due to the increase of the bottleneck link capacity, the energy conservation becomes less dramatic and it reaches the 5 Joule value (an observation in line with the results of scenario 1). When severe network congestion is present on the network TCP introduces considerable overhead in its effort to achieve reliable transmission. In the E2E case, the mobile node at the receiving end is involved in this extra effort and, consequently, does not frequently have the opportunity to switch to Sleep mode. Conversely, in the Split case the base station absorbs the effects of the network congestion (lost packets that lead to out-of-order data reception) and only transmits to the mobile host when consequent buffer sized chunks of data are available. In DTN terms, the base station will be receiving the incoming bundle without engaging the receiving node in the process. Upon successful bundle reception forwarding to the mobile node would start. In the meantime, the WNIC of the mobile host can switch to Sleep mode and save significant amounts of energy. The higher the congestion level the greater the energy gains achieved by the Split approach. In the case where PSM is not employed, the energy expenditure is again related to the transfer duration. In the 0.5Mb bandwidth setting the No PSM energy expenditure is twice as much as that of the Split case, dropping to about 50% more than the Split case in the 2.5Mb setting.

In the previous scenario we added cross traffic and rerun the experiments. In this set of experiments the maximum bandwidth value for the backbone link was increased to 3.5, since beyond that capacity the measurements remain unchanged.

L34 bandwidth (Mb)	Duration E2E (sec)	Duration Split (sec)
0.5	364.61	357.91
1	225.85	194.47
1.5	159.32	158.63
2	125.39	124.53
2.5	107.72	107.21
3	90.04	89.73
3.5	90.74	90.25

Table 3: FTP, Varying Bandwidth, Competing Flow Transfer Duration.

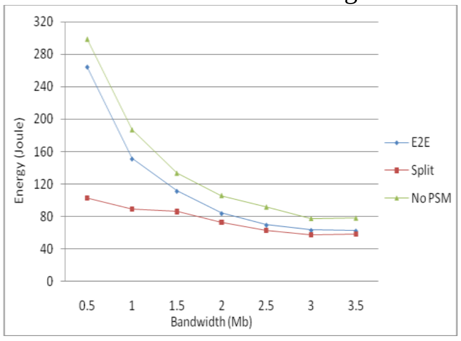


Figure 5: FTP, Varying Bandwidth, Competing Flow Transfer Duration.

As expected, the transfer duration, shown in Table 3, is significantly longer for both the E2E and the Split approaches when a competing flow is present. Nevertheless, unlike the previous scenario, small bandwidth values (i.e. 0.5 and 1 Mb) result to a slightly shorter transfer duration for the Split case.

Regarding the energy consumption, it can be observed in Figure 5 that despite the significantly longer time it takes for the transfer to complete for the 0.5Mb setting (around 60% more in both cases), in the E2E approach the additional energy requirement is 62%, while in the Split approach the additional energy requirement is only around 10%. As the bandwidth increases and the network is relieved from congestion, the energy conserved by the Split approach is limited to around 5 Joule (a value that was observed in almost all congestion-free experiments).

Figure 6 depicts the state transitions for the first 14 seconds of the file transfer in both the E2E and the Split cases when the backbone bandwidth is 1Mb. It is obvious from the chart that in the E2E case, the WNIC needs to switch to the Active state more frequently than in the Split case. For example, between 2 and 6 seconds, in the E2E case the WNIC needs to switch to the Active state 4 times, while in the Split case it remains in the Sleep state for the whole duration. Also, even when a switch to the Active state is necessary the time needed to remain in this state is shorter for the Split than for the E2E scenario (i.e. at 6, 7, 11 seconds).

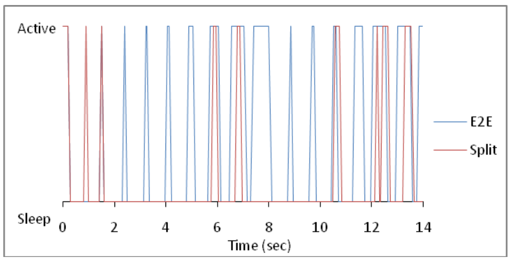


Figure 6: FTP, 1Mb Bandwidth, Competing Flow State Transitions.

5.3 Varying Wireless Error FTP Transfer

The purpose of this scenario is to reveal the effect of packet error on the wireless part of the network both on the transfer duration as well as the energy efficiency of the E2E and the Split approaches. The backbone link was assigned a 300ms delay and a 1Mb bandwidth. The delay was selected so that it is near the low end of the value range used in other experiment sets and the bandwidth was selected so that the backbone is also the bottleneck link of the transfer route (i.e. there is also network congestion present).

At first glance, it is striking that as the wireless packet error rate (PER) increases the duration of the E2E transfer increases as well (Figure 7), reaching in the 25% case more than three times the duration of the 0% case, whereas the duration of the Split case transfer remains almost the same across all runs. Part of the explanation to this, seemingly bizarre, result can be found in the Indirect-TCP solution proposed in [21]. In the E2E scenario, it takes very long for TCP to recover from packets lost due to the wireless error, because of the long RTT of the connection (around 800ms). On the contrary, the RTT of the wireless link is measured to be around 6-7 ms, and, therefore, a TCP connection between the base station and the mobile host can be very swift in recovering the lost packets. Additionally, since the RTT of the wired part of the route is very long, when compared to that of the wireless part, the time it takes for the base station to buffer the required data is enough for the TCP of the last hop to recover from any packet losses. This observation is not true only for the 30% PER case, in which it is evident that the last hop TCP cannot serve the incoming data, delaying the overall transfer time. In such a case, data could be dropped at the base station if the buffer capacity is depleted. We plan to include monitoring of packets dropped due to limited buffer capacity in future experiments. The performance gains of a solution along the lines of Indirect-TCP that isolates the wireless error in the short-delay part of the transfer route comes naturally as additional benefit when using DTN.

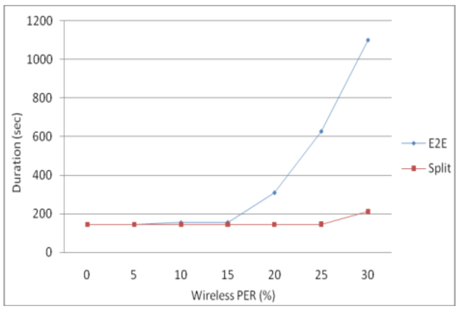


Figure 7: FTP, Varying Wireless PER, No Competing Flow Duration.

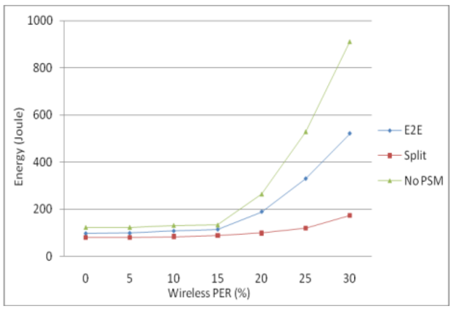
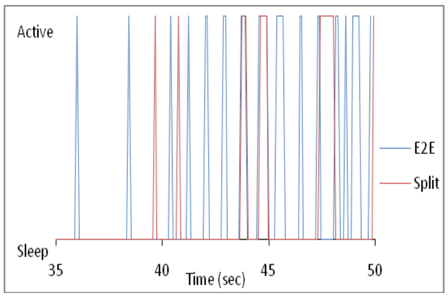


Figure 8: FTP, Varying Wireless PER, No Competing Flow Energy Consumption.

The energy consumption chart in Figure 8 shows that up to an error rate of 10% the energy savings achieved by the Split approach are approximately 16% - 20%. As the error rate increases so does the conserved energy, reaching as much as 66% for an error rate of 30%. At this error rate the energy required for the completion of the data transfer in the E2E case is 520.78 Joule, while in the Split case the required energy is 174.09 Joule. The energy expenditure in the No PSM case for the 30% error rate is as much as 910.49 Joule.

The state transitions for an indicative 15 second interval when the wireless PER is 20% are plotted in the chart of Figure 9. It can be observed that, for the whole duration covered in the chart, the WNIC switches to the Active state 14 times in the E2E case, while it only switches 6 times in the Split case. The state transition frequency for the two cases is similar during the rest of the data transfer, resulting in the significant energy expenditure difference observed in Figure 8 for a 20% PER.



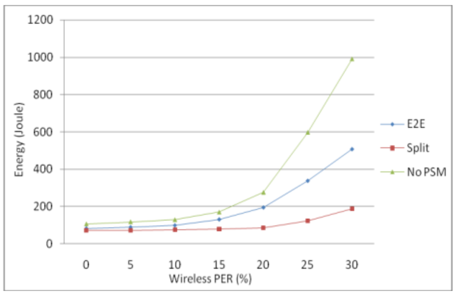


Figure 9: FTP, 1Mb Bandwidth, 20% Wireless PER, No Competing Flow State Transitions.

Figure 10: FTP, Varying Wireless PER, Competing Flow, Energy Consumption.

The experiments in the previous section were repeated with a competing flow present on the network. Because of the huge duration times that would otherwise result the bandwidth of the backbone link was set to 2Mb (as opposed to the 1Mb value that was used in the previous section). Even though we cannot directly compare the resulting figures for the cases with or without a competing flow we can still draw meaningful conclusions for the relative performance of the E2E vs. the Split case. The most significant observation in this scenario as depicted in Figure 10 is that when a competing flow is present, the energy expenditure of the E2E approach exhibit higher increase rates for smaller error rates. For higher error rates both the competing and the non-competing flow cases exhibit similar behavior.

5.4 Varying Buffer Threshold FTP Transfer

In this set of experimental runs the delay is set to 300ms, the bandwidth of the backbone link is set to 1Mb (rendering it as the bottleneck link of the transfer route) and the buffering size varies from 0 up to 200 KB. For the 0 buffering case we used the E2E approach, while for the rest of the runs we used the Split approach with the corresponding buffer size. The transfer duration is reasonably identical for all buffer sizes. The energy expenditure, however, drops significantly as the buffer size increases (Figure 11). Since the wireless link of the transfer route is consistently underutilized, buffering more data allows the WNIC of the receiver to switch to a Sleep state for longer periods, while at the same time not getting penalized for waiting. Increasing the buffer size beyond a certain value could cause congestion problems on the wireless LAN and/or prolong the transfer duration. Such problems could become more severe in case multiple mobile clients are present and download data using the Split approach. With the increase in the buffering capacity the traffic on the wireless LAN becomes burstier and consequently more prone to congestion. The size of the buffer, in case DTN is employed, can be thought of as the size of the bundle size that the base station would forward to the last hop of the transfer connection.

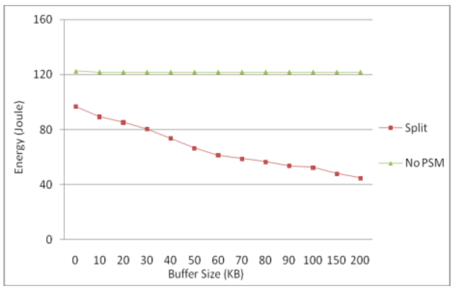


Figure 11: FTP, Varying Buffering Threshold, FTP, No Competing Flow, Energy Consumption.

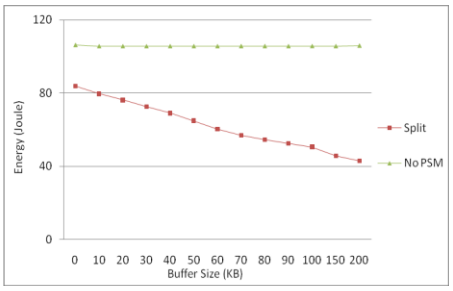


Figure 12: FTP, Varying Buffering Threshold, Competing Flow, Energy Consumption.

The experiments of the previous scenario were repeated with the introduction of cross traffic caused by a competing flow. In order to compensate for the higher network traffic the bandwidth of the backbone link was increased to 2Mb. In both sets of experiments the transfer duration remains constant for all runs. With respect to the energy expenditure, shown in Figure 12 it is observed that while for small buffer sizes the amount of energy required by the no competing flow scenario is significantly higher, as the buffer size increases the two values converge to a common Joule value in the low 40s. This shows that increasing the buffer size can alleviate differences in the energy expenditure caused by varying congestion conditions in the wired part of the network.

5.5 Varying Buffer Threshold CBR Transfer

In this last set of experiments a CBR traffic flow of 10MB is considered in a network with no contending traffic. As in most of the previous scenarios the delay of the backbone link was set to 300ms and the bandwidth to 2Mb. The transfer duration does not convey any important information as it merely refers to the time the last packet is received by the mobile node and it depends solely on the end-to-end network delay.

This scenario can be mostly considered as a placeholder for future work, since the side effects of delaying CBR traffic are not taken into account. When measuring the service quality of a file transfer the time for completion can be an adequate metric. The sooner the file transfer finishes the better it is for the end user. In all the experiments so far the transfer duration was not affected by the buffering mechanism, rendering the mechanism acceptable from a user standpoint. However, CBR traffic is, in most cases, related to multimedia streaming. Extra delays introduced by the buffering mechanism can lead to unacceptable service quality, a factor which is not captured by our current experimental setup. Nevertheless, the experiments in this scenario reveal the potential a DTN-based Split approach can have for reducing the energy consumption for CBR traffic.

It becomes apparent from the chart in Figure 13 that even for a buffer size of 20KB, the energy saving potential is 33% compared to the E2E case. As the buffering size increases the conserved energy increases as well, reaching almost 85% of the energy spent in the E2E case. Although large buffering in multimedia streams can cause long delays and, thus, unacceptable user experience, if used with caution it can deliver significant energy conservation with little or no quality degradation.

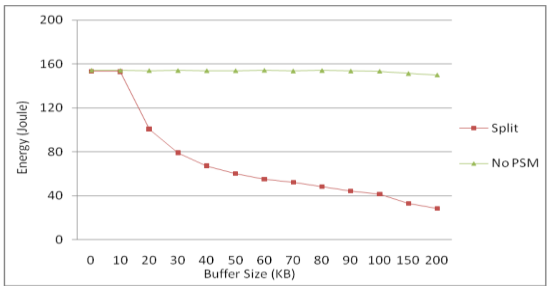


Figure 13: CBR, Varying Buffering Threshold Energy Consumption.

6. Conclusions

We evaluated the potential of DTN to shape Internet traffic in a manner that allows mobile devices to utilize energy-conserving modes of operation, without risk to lose packets or degrade their throughput. Our results are based on a post-simulation analysis of the recorded transition state diagram of the mobile device, as this was determined by the communication between the end-device and the edge network node. The most conclusive results of our experiments capture the huge gain in useless transitions and uncover the energy potential of DTN properties.

We briefly discussed the property of DTN to balance network traffic as well. This is not peripheral to energy-saving strategies: Load balancing within the network, unlike typical end-to-end approaches, cancels the risk of packet dropping but also forwards information towards the destination as soon as possible. Application data, instead of waiting at the source, it is pushed at some network nodes closer to the destination. This latter becomes particularly important in environments, such as space for example, where a contact graph is predetermined and the penalty for losing a contact opportunity may delay communication significantly. Hence, load balancing strictly associates with less retransmissions and shortened communication time: two tactics with energy-conserving consequences. We plan to focus on this dimension of energy-saving strategies by extending our present evaluation framework with load balancing algorithms.

Finally, taming uncertainty allows for more sophisticated signaling techniques that deploy a hybrid Time Division and Statistical Multiplexing strategy. This sort of sophistication is under progress.

References

[1] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, “Delay-Tolerant Networking Architecture”, RFC 4838, Internet Engineering Task Force, 2007.
[2] K. Scott and S. Burleigh, “Bundle Protocol Specification”, RFC 5050, Internet Engineering Task Force, 2007.
[3] M. Viredaz, L. Brakmo, and W. Hamburgen, “Energy Management on Handheld Devices”, ACM Queue Magazine - Power Management, Volume 1 Issue 7, 2003.
[4] A. Acquaviva, T. Simunic, V. Deolalikar, and S. Roy, “Remote power control of wireless network interfaces”, Proceedings of International Workshop on Power and Timing Modeling, Optimization and Simulation, 2003.
[5] C. Jones, K. Sivalingam, P. Agrawal, J. Chen, “A Survey of Energy Efficient Network Protocols for Wireless Networks”, ACM Journal on Wireless Networks, Volume 7, Issue 4, 2001.
[6] J. Adams and G.M. Muntean, “Adaptive-Buffer Power Save Mechanism for Mobile Multimedia Streaming,” Proceedings of IEEE International Conference on Communications '07, 2007.
[7] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, “Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications”, Proceedings of ACM/USENIX Internet Measurement Conference, 2009.
[8] L. Mamatas, V. Tsaoussidis, "Exploiting Energy-Saving Potential in Heterogeneous Networks", International Journal of Parallel, Emergent and Distributed Systems (IJPEDS), Taylor & Francis, volume 23, issue 4, 2008.
[9] A. Acquaviva, E. Lattanzi, and A. Bogliolo, "Design and Simulation of Power-Aware Scheduling Strategies of Streaming Data in Wireless LANs", Proceedings of

ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2004.
[10] J. Korhonen and Y. Wang, “Power-efficient streaming for mobile terminals”, Proceedings of ACM International Workshope on Network and Operating Systems Support for Digital Audio and Video, 2005.
[11] S. Chandra and A. Vahdat, “Application-specific network management for energy-aware streaming of popular multimedia formats ”, Proceedings of the General Track of the annual conference on USENIX Annual Technical Conference, 2002.
[12] G. Anastasi, M. Conti, and W. Lapenna, “A Power-Saving Network Architecture for Accessing the Internet from Mobile Computers: Design, Implementation and Measurements ”, The Computer Journal, Volume 46, Issue 5, 2003.
[13] H. Zhu and G Cao, “A Power-Aware and QoS-Aware Service Model on Wireless Networks ”, Proceedings of IEEE INFOCOM '04, 2004.
[14] H. Zhu and G. Cao, “On Supporting Power-Efficient Streaming Applications in Wireless Environments”, IEEE Transactions on Mobile Computing, Volume 4 Issue 4, 2005.
[15] G. Anastasi, M. Conti, E. Gregori, A. Passarella, and L. Pelusi, “A Power-Aware Multimedia Streaming Protocol for Mobile Users”, Proceedings of the IEEE International Conference on Pervasive Services '05, 2005.
[16] I. Batsiolas and I. Nikolaidis, “Selective Idling: Experiments in Transport Layer Energy Conservation”, The Journal of Supercomputing, Volume 20, Number 2, 2001.
[17] IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements , Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications , 2007.
[18] Eugene Shih, Paramvir Bahl, and Michael J. Sinclair, “Wake on Wireless: An Event DrivenEnergy Saving Strategy for Battery Operated Devices”, Proceedings of the ACM International Conference on Mobile Computing and Networking, 2002.
[19] T. Simunic and S. Boyd, “Managing power consumption in networks on chips,” ACM DATE, 2002.
[20] Yu Xiao, Petri Savolainen, Arto Karppanen, Matti Siekkinen, and Antti Ylä-Jääski, “Practical power modeling of data transmission over 802.11g for wireless applications”, Proceedings of the ACM International Conference on Energy-Efficient Computing and Networking, 2010.
[21] A. Bakre, B. R. Badrinath, “Implementation and Performance Evaluation of Indirect TCP”, IEE Transaction on Computers, Vol. 6, No. 3, 1997