Power-Aware Data Management for Small Devices

Sami Rollins and Kevin Almeroth
UC Santa Barbara
{*srollins,almeroth*}@cs.ucsb.edu

Dejan Milojičić
Hewlett Packard Laboratories
dejan@exch.hpl.hp.com

Kiran Nagaraja
Rutgers University
knagaraj@cs.rutgers.edu

ABSTRACT

Pervasive computing devices such as Personal Digital Assistants (PDAs) and laptop computers are becoming increasingly ubiquitous. The future promises even more advanced devices such as digital watches, jewelry, and even clothing. However, as pervasive devices become more widely used for more advanced applications, their resource limitations are becoming more apparent. In this work, we focus on data management and power limitations. We investigate the benefit of using power-aware schemes to automatically manage content across a collection of devices and prolong data availability. We monitor the available energy supply on each device and migrate content from devices that are in danger of dying. In our simulated environment, we have found that, using intelligent techniques for data management can increase the amount of time a collection of devices remains usable by over 2 times. Furthermore, our techniques can perform autonomously, independent of user intervention.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

design, performance

Keywords

energy, small devices, content management

1. INTRODUCTION

Devices such as hand-held Personal Digital Assistants (PDAs) and laptop computers are becoming increasingly ubiquitous. Users call upon them to manage their schedules, check their email, maintain their address books, and carry out their everyday work tasks. The future promises even more advanced devices such as digital watches, jewelry, and even clothing. Users will be able to have instant access to information using nearly invisible technology. The advent of such devices coupled with the ubiquitous deployment of wireless networking technology will enable users to create, manage, and access their own content as well as to seamlessly share information with colleagues and friends or upload and download information from infrastructure embedded in the environment. However, as we move toward a vision of completely mobile users with instant access to information using computers that have been cleverly and invisibly embedded into everyday objects such as clothing, we face a number of challenges.

In particular, as pervasive devices become more widely used, and as future generations of devices continue to promise to be smaller than the current generation, resource limitations of devices will have a greater effect on available services. These limitations include *user-centric* issues such as ease of management of multiple devices. They also include *resource-centric* constraints of CPU, bandwidth, memory, storage, and especially power.

In this work, we focus on data management and power limitations. We investigate the benefit of using power-aware schemes to automatically manage content across a collection of devices. In this context, managing content includes migrating content to the device where it is likely to be accessed. We make the assumption that a user is likely to carry, not one, but a collection of devices. By viewing the collection as a single unit, and managing the resources available across the collection, we can increase the amount of time that the collection remains usable. Our evaluation of this idea focuses on quantifying the benefits in a simulated environment.

We simulate a data management strategy that monitors the remaining lifetime of each device in a user's collection. When the lifetime of a particular device gets critically low, the other devices in the collection are polled for their remaining lifetimes. Content (i.e., data files) the user is likely to access is migrated to the other live devices that still have a reasonable remaining lifetime. This enables a user to continue to work in the event of complete power loss on a particular device. We have found that, using intelligent techniques for data migration can increase the amount of time a collection of devices remains available by over 2 times. Our techniques have user-centric implications as well. Users can take advantage of longer device availability without having to explicitly distribute tasks among devices.

This paper is organized as follows. In Section 2 we discuss current uses of small devices, motivate the problem of limited resources, and discuss related work. In Section 3 we introduce the idea of power-aware data management and present the specifics of our algorithm. We present our simulations and evaluation of data management in Section 4. We conclude in Section 5.

2. MOTIVATION

The goal of this work is to explore power-aware data management across small devices. We argue that pervasive devices often

WoWMoM'02, September 28, 2002, Atlanta, Georgia, USA.

have resource limitations that can be overcome by using a collection of devices in a cohesive way. However, using the available resources across a collection of devices can be a challenge. In this section, we discuss our target devices and applications, the limitations of these devices, and current approaches that have partially addressed these limitations.

2.1 Devices and Applications

Pervasive computing devices range from powerful laptop computers to information appliances to computing devices embedded in everyday objects. In this work, we focus primarily on small, mobile, user-centered computing devices. Current examples of these devices include laptop computers, PDAs, and cellular telephones. In the future we imagine other devices such as digital watches or rings, other jewelry, or clothing [12]. In short, our focus is on devices that a user might carry throughout a day. Although we do not focus on devices such as embedded sensors that are centered around the environment rather than the user, we believe that many of our techniques may be applicable there as well.

As devices become more advanced, the range of applications they support is ever increasing [19]. In particular, increased disk space, memory, and bandwidth have enabled small devices to be used for more advanced, content-driven applications. While the first generation of PDA devices supported tasks such as calendar and address book maintenance, newer PDAs support access to text documents and multimedia content such as audio and video files. In the next generation, the same functionality may be supported by a user's jacket (http://www.media.mit.edu/hyperins/levis/). Moreover, increased connectivity between devices and between users has promoted more advanced content sharing applications. Today, we can imagine users on a subway train exchanging sections of the morning newspaper [14]. In the future, we can imagine millions of computers embedded in the environment, each gathering information and making sure the information is available where it is likely to be accessed.

Managing information in such a large-scale environment promises to be a great challenge. In fact, even in a more limited environment challenges arise. As devices become smaller, a single user is more likely to carry a collection of devices. Even today we see business people who carry a PDA to keep track of appointments, a cellular phone to speak with colleagues, and a laptop computer to aid in presentations and perform other complex chores. A high school student might carry an MP3 player to listen to music, a cellular phone to send messages to friends, and a laptop computer to take notes in class. In the future, the same student may be wearing digital earrings to listen to music, a digital shirt to capture her voice and transmit her instant voice message to friends, and a digital jacket with a keypad to allow her to take notes in class.

To be more specific, imagine a user on an airplane. She picks up her laptop and starts to modify a document. When the laptop runs out of battery, she picks up her PDA and continues to modify the document using the more limited interface. Eventually, the PDA runs out of battery and she switches to her cell phone. Finally, when her cell phone runs out of battery she resorts to her digital watch where she can dictate text using speech recognition. Despite the small number of devices, managing content in this scenario is particularly challenging, primarily because pervasive devices have, and will continue to have, resource limitations.

2.2 Device Limitations

Devices such as PDAs and laptops are becoming increasingly powerful. The most powerful laptops on the market today rival desktops of only a few years ago in terms of processing speed, memory, disk space, and bandwidth. However, battery technology is not advancing at near the rate as other resources [5]. Often, a single battery will only sustain a laptop for a couple of hours or less under a normal workload.

As devices become more ubiquitous, the power limitation becomes even more of a burden. Not only will there continue to be situations where power is simply not available, even in locations where power is available, it may be limited. For example, at a disaster site where recovery workers need to store and communicate information they have gathered, power is unlikely to be available at all. Alternatively, at an airport where power might be available, there is no guarantee that an outlet will always be free.

While it would seem intuitive that a user who carries a collection of devices would be able to accomplish more than if the user carried a single device, managing and using the collection is often difficult. Despite redundant functionality, multiple devices often cannot be used to accomplish the same task. The primary problem is lack of coordination. Especially with content-driven applications, if the content a user wants to access is not available on the device where the user is trying to access it, the task cannot be completed. This is particularly challenging in a power-limited environment. A user may have a PDA and a laptop computer that both have a text editing program installed. However, if a user is composing a paper on a laptop and the laptop battery dies, unless the user has manually copied the file, she cannot continue to work on her PDA.

As the practice of carrying multiple devices on a day-to-day basis becomes more commonplace, creating tools for automatically managing and coordinating them becomes necessary. The ideal situation from the user perspective is to accomplish *any task on any device*. For example, a task that a user would normally perform on a laptop should still be possible, even if the laptop's battery has died. A generic framework for managing and coordinating devices should monitor available resources, and perform tasks based upon knowledge of the complete system. Resources can include CPU, network bandwidth, disk space, and battery lifetime. In this work, we focus specifically on strategies for managing data across a collection of devices in a power-aware way. Our goal is to intelligently monitor power and migrate data such that it remains available even in the face of a power loss on a particular device.

2.3 Related Work

The concept of managing resources across devices has been looked at from a variety of angles. The MOPED project [9] proposes aggregation of a collection of devices from the network layer by using well-connected gateways to communicate on behalf of other devices. While MOPED primarily seeks to integrate the devices belonging to a single user, a similar project has looked at aggregating connectivity across a larger collection of devices [13]. From the user perspective, the MPA project [16] addresses aggregation by using an infrastructure that tracks users and chooses the *best* device on which to contact them. Finally, Roma [22] manages data stored across a collection of devices by storing metadata about all data on a single, portable device. In this work, we are primarily interested in using power-aware techniques to increase the amount of time useful data remains available across a collection. We address this challenge primarily from the resource point-of-view, not necessarily the user point-of-view.

From the resource point-of-view, a number of projects have looked at *conserving* the resources of small devices. The resource most often targeted is power. A number of projects have looked at conserving power across devices from the application point-of-view [1, 4], the CPU point-of-view [18], the memory point-of-view [11], and the network point-of-view [10, 20]. Power conservation

has also been addressed from the point-of-view of ad hoc routing among many small devices [6, 8]. However, *conserving* power may not be sufficient in many situations. Eventually, a device running the most conservative power saving schemes will run out of battery power. In this work, we focus on extending the availability of data by intelligent use of power resources across a set of devices. This can be used alone or in concert with conservation schemes.

3. STRATEGIES FOR POWER-AWARE DATA MANAGEMENT

In this work, we explore power-aware management of data across a set of pervasive devices. Our strategy focuses on viewing the collection of devices as a single unit and managing data available across them using power-aware schemes. Using this strategy, we can increase the amount of time that useful data remains available, even in the face of a power loss on a subset of the devices. In this section, we look at an overview of the goals and idea behind our management scheme and then discuss our design in more detail.

3.1 Overview

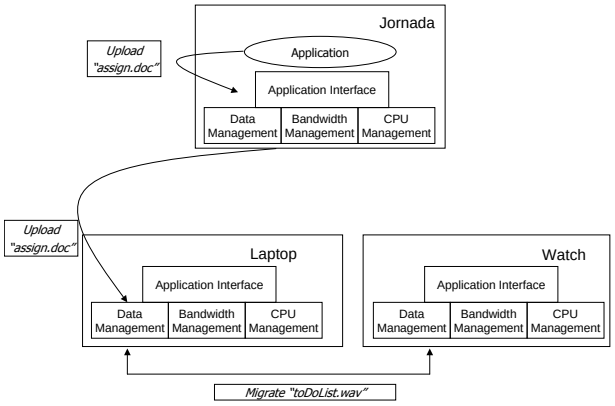

Figure 1: An overview of our architecture.

Figure 1 illustrates a high-level view of our architecture. A user carries a collection of devices (e.g., a Jornada, laptop, and digital watch). Each device runs a middleware component that is configured such that it knows about the other devices in the collection and can communicate with them. When one device receives a request to perform a task, such as uploading a file, that device can divide the task into subtasks and allocate the subtasks to the appropriate devices in the collection. For example, a device may receive a request to upload a file that is not stored locally. In that case, the device would pass the request to the device that it knows has a copy of the file.

This model can apply to a variety of resources. From the CPU point-of-view, a compute intensive task can be offloaded to the device with the most processing power. From the bandwidth point-of-view, the task of downloading a piece of content might be transferred to the device with the most available bandwidth. Our focus is on data and power. In particular, we focus on maintaining availability of data. Such a strategy can include elements such as coordinated data upload/download. However, the main focus of our evaluation is on maintaining data availability in a power-constrained environment.

Figure 2 illustrates an overview of the use model we assume. Each device in a collection stores some set of files. When a file is requested from one device, even if the file is not currently available on that device, the request is serviced by redirecting the request, or fetching the requested file on-demand. Moreover, even if the file was originally on a device that has lost power, the file should still be available.

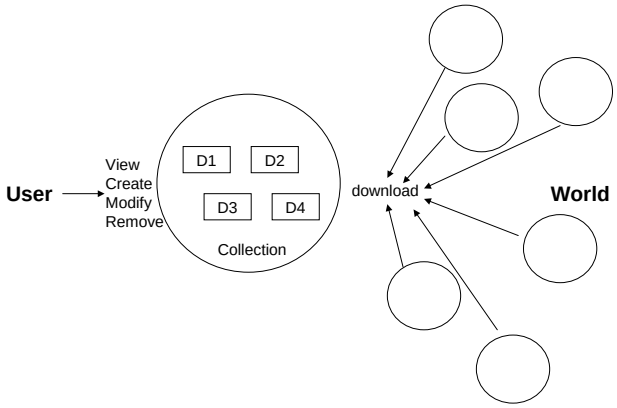

Figure 2: An overview of our model for use.

In the most common case, requests will be issued by the user. A user may want to view, create, modify, or remove content from her devices. To provide this support, a data management component should act as a distributed file system. A user should have a complete view of the files available across the collection. If a requested file is not on the device in use, it should be fetched, on-demand, from its current location. To implement such a scheme, each device stores metadata about the files stored on the other devices. Alternatively, we could employ a more centralized model such as in the Roma project [22]. The metadata is updated when changes occur. When a request is made, the metadata is consulted and the file is fetched from the appropriate location if necessary.

Similarly, such a scheme can support content exchange with other users. A collection of user devices may act as a server, providing content to other users that are within wireless range of the collection. For example, a group of users in a public place like a mall or a sporting event may form an ad hoc network and wish to share data in a peer-to-peer fashion. A request for download made to a particular device in the collection should either be serviced by that device, or rerouted to the device that has the content. Alternatively, the environment itself may provide a collection of devices and information can be downloaded from or even uploaded to the resource-rich infrastructure. In this context, other challenges such as peer discovery, group membership, and data location arise. Numerous current research efforts are addressing these challenges [15, 21, 17], therefore we do not address them here.

The main goal of our work is to maintain availability of data in a power-constrained environment. Regardless of whether the user or the outside world is requesting data, we want to increase the probability that the requested content is accessible. To meet this goal, we monitor the power supply available across the collection of devices, and migrate data from those devices with a low power supply to those devices where the data will be most useful.

3.2 Implementing Data Management

To implement data management, each device in the collection must run a data management component that performs migration. The component behaves according to the following algorithm:

```
poll the operating system for the remaining energy level
if the level is below a threshold
  migrate subset of items on this device to 1 or more
    target devices in the collection
repeat
```

This algorithm is quite general and leaves a number of questions unanswered. The first question is how we determine the threshold for migration. The threshold is determined by taking into account the remaining energy on the device, the amount of content to be migrated (e.g., all content in a specified directory), the network throughput of the device, and the power consumption characteristics of the device. We assume that the power consumption and network characteristics can be configured, or can be dynamically determined by observing system behavior. When the amount of energy remaining is equal to the amount of energy it will take to migrate all of the required content, migration is initiated.

Another question is what happens if a user has picked up a particular device and taken it out of range when migration is initiated. In this case, a potential target is unavailable. As long as one live device is within range, migration can occur. However, if no devices are within range, all data on the dying device becomes lost. A potential solution to this problem might be to incrementally migrate content throughout the life of the device. Another option might be to monitor the wireless range of the collection of devices and perform migration when a device starts to go out of range. Both solutions are likely to incur some additional overhead in messaging between the devices in the collection. We leave further exploration of this problem as future work.

The next question addresses the strategy for choosing which data should be migrated from a dying device. Migrating too little content may not be helpful while migrating too much content is wasteful of the resources required to transmit content. Because the answer to this question is not straightforward, we investigate three strategies for choosing which data to migrate:

Migrate All (MA) - In a migrate all strategy, all data is migrated from the device losing power. While it may not be practical to migrate *everything* on a particular device, this strategy can be implemented by migrating all data that is located in a specified directory.

Migrate Most Recently Used (MRU) - In a most recently used strategy, all data that has been accessed by the user recently will be migrated. This strategy assumes that a user is likely to access some specific subset of the available content at a given time. For example, a user might be accessing all documents related to one specific project.

Access Optimized (AO) - In an access optimized scheme, we assume that we have some knowledge of which data the user is likely to access. Either the user can specify a list of content that is likely to be accessed, or we can implement some form of user profiling.

The final question addresses the strategy for choosing the destination device(s). We can imagine a number of very simple schemes such as round-robin or simply choosing the device with the largest remaining power supply. However, such schemes may be ineffective, or wasteful. Using a round-robin scheme, content may be migrated from a dying device to another dying device. In this case, the content will only have to be migrated again, likely before it is even used. Choosing the device with the largest remaining power supply may also be ineffective if the target device has very high energy usage characteristics. Additionally, considerations such as available storage space or bandwidth of the target device are also likely to have an effect on performance.

In this work, we evaluate a **Power Aware (PA)** scheme for choosing the destination device. The amount of content migrated to a particular device is based on the remaining lifetime of that device with respect to the lifetimes of the other devices. We poll each live device *d* in the collection for the amount of time it can continue work before it reaches its threshold (rl_d). For each device *d*, the number of items migrated to *d* with respect to the total number of items *(ni)* is $(rl_d * ni)/\sum_{i=1}^{N} rl_i$. Similarly, we could consider the total size of the items instead of the number of items. However, for the purposes of our experiments, we assume that all files are roughly the same size. This formula ensures that the amount of content migrated to a given device is proportional to the remaining lifetime of the device with respect to the remaining lifetimes of the other devices in the collection. The goal of this scheme is to ensure that content is available on devices that are likely to be alive when the user tries to use them.

We could also imagine even more advanced schemes such as predicting the remaining energy supply *after* migration occurs, or predicting the device where the user is most likely to access the specific content. In the first case, implementing such a scheme would only be necessary if the ratio of energy consumed to remaining energy supply varied greatly across the collection of devices. The second option would avoid the case of migrating content to a device, and then having to fetch it, on-demand, from another device before it can be used. However, to implement such a scheme, we need to either explore user profiling techniques, or require the user to specify where content is likely to accessed. We leave this question as future work.

4. EVALUATION

In this section, we evaluate the data management schemes introduced in Section 3 using simulation. We first discuss the use model we assume. Then, we outline the metrics we evaluate and the setup of our simulation. Finally, we present our results.

4.1 Simulation Overview

In this paper, we evaluate data management from the *user* perspective. In this section, we detail the set of assumptions we make about user behavior.

First, we assume that a typical user carries four devices: one large device such as a laptop; two medium sized devices such as a PDA and a cell phone; and one small device such as a digital watch. Choosing a different set of devices would have limited effect on the results of our simulations since each type of device we model has roughly the same lifetime given the ratio of full energy to energy consumed when active. Further, we assume all devices have similar functionality and can be easily interchanged for the tasks we evaluate.

Next, we assume that a user uses one device at a time, and uses that device until the battery has been exhausted. To clarify, we revisit our example from Section 2. Imagine a user on an airplane. She picks up her laptop and starts to modify a document. When the laptop runs out of battery, she picks up her PDA and continues to modify the document using the more limited interface. Eventually, the PDA runs out of battery and she switches to her cell phone. Finally, when her cell phone runs out of battery, she resorts to her digital watch where she can dictate text using speech recognition.

Finally, we assume that one of the devices is constantly in use, and the remaining devices are idle (or have run out of power). In practice, an intelligent power saving scheme that puts all devices into sleep mode when no device is used is likely to be employed. We assume such a scheme is used, therefore we do not simulate intermittent periods when all devices are idle. As described in Section 4.5 we could employ additional power saving techniques to reduce power consumption of the idle devices when one device is in use. However, we leave this question as future work. For the purposes of this work, we assume that all unused devices must be on and idle such that any content not available on the device currently in use can be fetched on-demand. That is to say, if a user wants to listen to an MP3 stored on her PDA while she is currently using her laptop, the file will be fetched, on-demand, from the PDA.

In our simulations, we model user behavior by creating a trace of items that will be accessed. Iterating through the set of devices, we model a user picking up a device and accessing the next piece of content in the trace for a specified period of time. The trace of items is finite. If the current piece of content is not available on the current device and cannot be fetched from another device (i.e., the only device storing the content has run out of battery power), we move to the next piece of content specified in the trace. When the current device reaches its power threshold, migration is initiated. As described in Section 3, all remaining, reachable devices are polled, and content is migrated to the most appropriate device(s).

4.2 Metrics

Our evaluation looks at three main metrics:

Time Devices Remain Usable - The primary metric we are interested in is the total time that a user can work. If the user is interested in accessing data from a device that has gone dead, even if another device remains available, the user cannot make progress. By migrating relevant data to devices that are still available, we hope to increase the total time the set of devices remains usable.

Data Misses - A secondary metric is the number of misses the user experiences. A miss is characterized by the user picking up a device and attempting to access a piece of data that has been migrated to another device, or was not migrated from a device before it died. Misses are important because they indicate that the data migration strategy has failed to migrate the data to the appropriate device. Some misses can be overcome by migrating data from another device on-demand. However, if we fail to migrate data from a device that has died, the user cannot complete the current task and will have to attempt a new task.

Wasted Migrations - The third metric is the amount of power that is wasted by migrating data that is never accessed on the new device. Migrating data requires additional power that would not otherwise be consumed if a user were simply working on the device. Therefore, wasting power reduces the total amount of time a user can work on a particular device.

4.3 Setup

For our simulations, we modeled four devices; a large device such as a laptop computer, two medium devices such as a PDA and a cell phone (with the same characteristics), and a small device such as a watch. Our small device characteristics (see Table 1) are loosely based on measurements reported by Hill et al. [7]. While we are not aware of a small device on the market that achieves the functionality we intend, we believe that such a device will exist in the future.

The large and medium device characteristics we use are taken from measurements reported by Farkas et al. [2]. They measured power consumption of the IBM ThinkPad 560x running at 233MHz using 64Mbytes of memory and the Itsy Pocket Computer with a SA-1100 processor running at 132.7 MHz using 64Mbytes of DRAM and 32 Mbytes of flash memory. We use the idle, busy wait, and busy wait with LCD-enabled benchmarks. While different workloads may consume different amounts of energy, differences are often slight and these measurements are reasonable for the granularity of our experiments.

Because Farkas et al. and Hill et al. do not measure the power consumption of a network interface, we use the measurements taken by Feeney and Nilsson [3]. They measure the consumption of a 2.4GHz DSSS Lucent IEEE 802.11 WaveLAN PC Silver card (11Mbps) running on an IBM ThinkPad 560. To obtain the device characteristics shown in Table 1, we have added the measurements reported by Feeney and Nilsson (idle, receiving, and transmitting) to the measurements reported by Farkas et al. and Hill et al. This provides us with a reasonable model for our simulation. Finally, the full energy supply for the medium device is based on the battery type specified by Farkas et al. High-quality AAA batteries have a capacity of 700 mAH. The large device supply is taken from numbers reported by Flinn and Satyanarayanan [4].

A device can be in one of six states:

Idle - In the idle state, all components of the device are in idle mode including the display and network card.

Active - In the active state, the processor and display are busy, and the network card is idle. This state models the user accessing content on the device without having any network interaction.

Inactive Receiving (I/R) - In the inactive receiving state, the processor is busy, the display is idle, and the network card is in receiving mode. This state models the device receiving the content migrated from another device. However, the receiving device is not being used by the user.

Active Receiving (A/R) - In the active receiving state, the processor and display are busy, and the network card is in receiving mode. This state models the device receiving the content migrated from another device while the user is accessing content on the receiving device.

Inactive Transmitting (I/T) - In the inactive transmitting mode, the processor is busy, the display is idle, and the network card is in transmitting mode. This state models migrating content from a device while the user is not accessing content on the particular device.

Active Transmitting (A/T) - In the active transmitting state, the processor and display are both busy, and the network card is in transmitting mode. This state models migrating content from a device that the user is currently using to access content.

A device transitions into an *active* state if the user attempts to access content on the device. A device transitions into a *receiving* state if content is migrated to the device, or if the device fetches content from another device on-demand. Finally, a device transitions into a *transmitting* state if the device is migrating content to another device, or is responding to an on-demand fetch.

We assume a fixed amount of storage is initially used on each device. For the large device, we assume 3.5Mbytes, for the medium devices, 2.0Mbytes, and for the small device, we assume 256Kbytes. Experience using other values for the initial storage used shows that varying this parameter does not affect the results. Moreover, we do not place any restrictions on the maximum storage available on each device. Instead, we compare multiple migration schemes and evaluate the wastefulness of each with respect to storage used across the collection of devices.

We vary the following characteristics:

Number of items stored - While the amount of storage used remains stable, we vary the number of items initially stored on each device such that the size of each item varies. We evaluate two schemes: (1) **FEW** where the large, first medium, second medium and small devices store 12, 10, 8, and 2 items respectively and (2) **MANY** where the large, first medium, second medium, and small devices store 24, 20, 16, and 4 items respectively. A user who stored many papers or presentations on her devices might reflect the **FEW** model whereas a user who stored a cache of email messages or web pages might reflect the **MANY** model.

Time for each access - We evaluate two patterns for the amount of time each piece of content is accessed. In the **SHORT** pattern, each piece of data is accessed from 0-50 minutes. A user who is listening to music files, or browsing through cached web pages might follow the **SHORT** access pattern. In the **LONG** pattern, each piece of data is accessed from 50 minutes to 2 hours. A user

Size	Full Energy Supply (J)	Idle (W)	Active (W)	I/R (W)	A/R (W)	I/T (W)	A/T (W)
Small	2000	.01	.28	.32	.35	.4	.5
Medium	7560	.86744	1.186	1.3126	1.3476	1.75815	1.79315
Large	90000	3.93944	13.839	9.2006	14.0006	9.64615	14.44615

Table 1: Power consumption characteristics of devices simulated.

who is working on a paper or presentation would likely follow the **LONG** access pattern.

Trace distribution - We compare two distributions for generating the trace of content a user wishes to access. The first distribution is **Zipf** [23] which we feel is reasonable for many applications. Zipf models the case when a user primarily accesses a small subset of the entire available content base. We also compare the Zipf distribution against a **uniform** distribution.

Throughput for each device - We assume that each device has an 11Mbps wireless card. However, in practice, wireless throughput tends to be much less than 11Mbps. Thus, we examine various values for the throughput achieved.

Migration scheme employed - We compare six migration schemes. In the first scheme, **NONE**, no migration is employed. In the second scheme, power aware migrate all (**PAMA**), we use a power aware scheme for choosing the destination device, and we migrate all data from the current device. In the third scheme, power aware most recently used (**PAMRU**), we use a power aware scheme for choosing the destination device and we use a most recently used scheme for determining which data to migrate. In the most recently used strategy, we do not assume any history is kept from session to session and all data migrated is that which has been accessed since the simulation began. In the fourth scheme, power aware access optimized (**PAAO**), we look ahead in the trace of items to model user preference. In essence, we assume that we have perfect knowledge of the documents the user will access and migrate only those items that appear in the remainder of the trace. In the fifth scheme, ideal access optimized (**IAO**), we assume that all data is already in its ideal location such that no migration is necessary and a user can always continue working. In the final scheme, ideal power and access optimized (**IPAO**), we make the same assumptions as in the IAO scheme. However, we further assume that any device not in use can be safely turned off. Thus, this scheme wastes no power in idle mode.

4.4 Results

In Figure 3, we vary the migration scheme, number of items stored, and length of time for each access for the Zipf trace distribution. The results presented illustrate the amount of time that a user successfully accesses some piece of content (i.e., gets some work done). Our first observation is that intelligent migration schemes substantially increase the time a user can access information. In the LONG/FEW case, the PAMA scheme increases the duration by more than 2 times over NONE. In fact, PAMA is almost as efficient as PAAO, the user preference scheme.

Our second observation is that our migration schemes approach, and in some cases *exceed* the IAO scheme. IAO measures the amount of time a user can complete work if she uses each device, one at a time, until all power is exhausted on that particular device. The remaining devices are on and idle during the entire experiment. We were initially surprised by the result that our migration schemes could perform better than IAO. However, closer inspection of the behavior of the system led us to the conclusion that the reason we perform better than IAO is because of the power consumption characteristics of the small device. The small device consumes less power, and hence has a longer overall lifetime than the other devices. Thus, in our schemes we waste power migrating, and exhaust the power on the large and medium devices sooner than in the IAO case. The result is that we end up using the small device for a longer period of time and can take advantage of the fact that it consumes only a small amount of power while active. In the IAO case, the other devices can be used for a longer period of time, during which the small device's lifetime is reduced by consuming idle power. The conclusion we can draw from this observation is that the order the devices are used in has an effect on the overall lifetime of the collection when the ratio of full energy to active power consumption is high. In the future, we plan to explore the behavior of a scheme where external users request download of content and content is retrieved from the device where the content is stored. This will give us a better understanding of the behavior of migration when devices are accessed in random order.

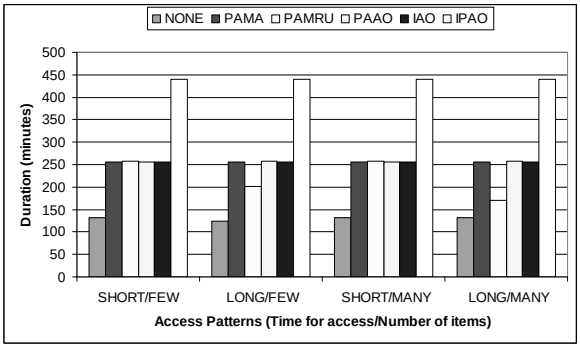

Figure 3: Total time devices remain usable for Zipf data access pattern.

Our third observation is that behavior is reasonably consistent across access patterns indicating that migration would be useful for a variety of applications. One thing we do notice is that PAMRU performs worse when the time for access is LONG. This can be explained by the fact that LONG accesses mean that fewer items are ultimately accessed. The result of this is that a smaller history is built and less information is available about most recently used items. We have performed some initial experiments to determine the usefulness of keeping a longer history, but have observed inconsistent results. Essentially, the shorter the history, the more unpredictable the behavior. The longer the history, the more PAMRU models the PAMA scheme.

Our final observation is that our strategies fail to approach IPAO. This is because the power conservation of our devices during idle time is quite high. In order to compete with an IPAO scheme, we need to employ intelligent power saving techniques such as turning off the network card or powering down unused hardware on the idle devices. Section 4.5 provides a brief discussion of the challenges of integrating these strategies. We leave further details as future work.

In Figure 4, we evaluate the same parameters as in Figure 3, but for a uniform trace distribution. Much of the behavior is similar to that of Figure 3 indicating that our schemes are reasonably effective

for multiple trace distributions. However, we do notice that, for the uniform distribution, NONE performs well and even outperforms our migration schemes for the SHORT access pattern. The reason we see such an improvement in NONE is because, in a uniform distribution, requested content is more likely to be stored on, and fetched on-demand from, all of the devices. In the event of a failure of a particular device, the likelihood that the user can find an alternate piece of content is high. In the SHORT/MANY case, NONE outperforms migration because no energy is ever wasted transferring content between devices. From these observations, we can conclude that migration is unlikely to be effective for applications in which the user can make use of virtually *any* piece of content stored on any device. However, this is unlikely to be the case for most applications.

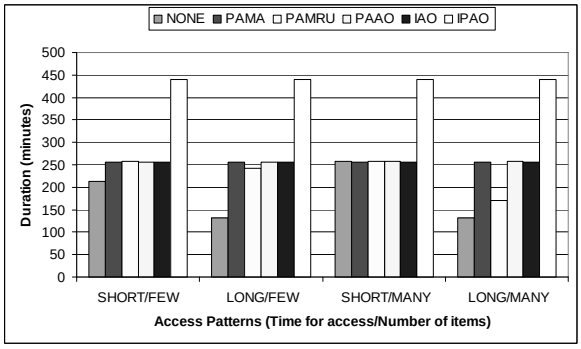

Figure 4: Total time devices remain usable for uniform data access pattern.

In Table 2, we take a closer look at the number of items that are migrated and never accessed at the new device (Wasted Migrations) and the number of times a piece of content must be fetched on-demand (Fetches) or cannot be accessed because it was not migrated from a device that died (Total Misses). We do not include the ideal schemes, or NONE because no migrations occur in either of the three schemes. The results presented are for the Zipf trace distribution, but the results for the uniform distribution were similar.

Our first observation is that PAMA wastes quite a lot of energy in all four access patterns. This is not surprising since PAMA essentially implements a fully replicated file system across all devices in a collection. In some cases, this may be an acceptable solution. In fact, we see that full replication does not have a substantial impact on the overall availability of the collection in the SHORT cases. But, full replication not only impacts the energy spent transferring content, it also impacts the storage space used on the collection of devices. Replicating all data onto a watch may not be feasible. Therefore, the tradeoff is to use a scheme, such as PAMRU, that incurs the penalty of increased Total Misses but reduces migrations. For some applications, the percentage of Total Misses may be unacceptable. However, for many applications, a user can *browse* the content the is available and take advantage of a wider range of files. It is noteworthy to point out that the trace of requests we generate is finite, but slightly exceeds the total amount of time the collection of devices will remain available. There are two impacts of this. First, this is why the PAAO scheme experiences a non-zero number of Wasted migrations. Also, since we assume that a user's list of tasks exceeds the amount of time she will actually have to complete them, even if a Total Miss is experienced, the user may be able to continue working. Making the list of tasks longer can increase the time work can be done in both the NONE and PAMRU schemes.

Data Access Pattern	Scheme	Throughput (Kbps)	Duration (minutes)
SHORT/ FEW	PAMA	11,000	256.68
		500	252.83
		250	248.78
		100	236.67
		56	220.9
	PAMRU	11,000	256.82
		500	255.83
		250	254.8
		100	251.68
		56	247.63
LONG/ MANY	PAMA	11,000	256.68
		500	253.03
		250	249.18
		100	237.7
		56	221.23
	PAMRU	11,000	170.05
		500	170.05
		250	170.05
		100	170.05
		56	170.05

Table 3: Total time devices remain usable for varying throughput rates.

In the results we have presented thus far, PAMA has performed very well in terms of usable time despite the overhead incurred by migrating so much content. A primary reason for this result is that the throughput of the devices is quite high at 11Mbps. In a wireless environment, using a CPU constrained device, 11Mbps throughput is likely not achievable. Collision in the wireless medium as well as the constrained processing power of devices drastically reduces throughput. In Table 3 we compare usable time for PAMA and PAMRU with reduced throughput. As we expected, the usable time for the PAMA scheme decreases with decreasing throughput while the PAMRU scheme remains relatively stable. This is simply because a slower connection will result in longer transfer times, and more power consumed during transfer. What we do notice is that from 11Mbps to 500Kbps, there is little difference in the usable time. Further, below 500Kbps, there is some drop, but the drop is not substantial. This is encouraging since it indicates that the overhead of migration is unlikely to be impacted by the throughput of the device. Though, the final observation we can make is that the transfer overhead will clearly increase if data objects are larger.

4.5 Discussion

The results that we have presented in this section are all quite promising. Migration performs well using a variety of devices working under a variety of workloads. However, there are a few exceptional cases where migration is not as beneficial. First, if the idle power consumption of the devices used is not substantially less than the power consumption in active mode, all devices are likely to run out of power at the same time. To improve this situation, we could integrate power saving strategies such as turning off the network card, or putting the entire device to sleep during idle time. To integrate such as scheme, we could simply periodically wake the sleeping device and poll the remaining devices to determine if any content must be migrated. Since we already employ batched migration, changes to our algorithm would be minimal. Primarily, we would need to be able to initiate migration before the threshold is reached if another device is alive and likely to be asleep again when the threshold is reached. Alternatively, we could rely on the user to wake sleeping devices if notified that migration needs to occur. However, this would be a less preferable option. The other challenge is determining the power available on the remaining devices

Data Access Pattern	Scheme	Total Migrations	Wasted Migrations	Access Attempts	Fetches	Total Misses
SHORT/FEW	PAMA	32	24	14	2	0
	PAMRU	5	1	23	2	11
	PAAO	15	7	14	2	0
SHORT/MANY	PAMA	63	55	14	3	0
	PAMRU	6	3	32	2	19
	PAAO	22	14	14	3	0
LONG/FEW	PAMA	32	29	6	2	0
	PAMRU	2	0	13	1	9
	PAAO	7	4	6	2	0
LONG/MANY	PAMA	64	60	6	4	0
	PAMRU	4	2	13	3	8
	PAAO	11	7	6	4	0

Table 2: Misses and wasted migrations for Zipf access pattern.

for the power-aware scheme. To solve this problem, we could simply cache the power available on each device since it will change very little while the device is in sleep mode.

In certain cases our algorithm may choose to migrate large amounts of data (e.g., an MPEG movie) that the user is not interested in accessing, or may fail to migrate data on which other data may be dependent. We argue that these exceptional cases could be avoided by allowing the user to *tag* data or specify characteristics of data that should not be migrated or should be migrated together.

5. CONCLUSION

In this work, we have investigated the benefit of using power-aware strategies for managing data across a collection of devices. Our results show that by monitoring available lifetime and migrating critical data, we can increase the usability of a collection of devices by over 2 times. Additionally, we observe that the overhead of such a scheme is relatively low compared to the benefit. Our strategy remains hidden from the user while transparently ensuring that a user can continue work even if a device runs out of battery. As pervasive devices become increasingly popular, developing and deploying these kinds of techniques will help to make devices both more powerful and more usable.

6. REFERENCES

[1] C. Ellis. The case for higher-level power management. In *HotOS*, pages 162–167, Rio Rico, AZ, USA, Mar. 1999.

[2] K. Farkas, J. Flinn, G. Back, D. Grunwald, and J. Anderson. Quantifying the energy consumption of a pocket computer and a java virtual machine. In *Sigmetrics 00*, pages 252–263, Santa Clara, CA, USA, June 2000.

[3] L. Feeney and M. Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *INFOCOM 01*, pages 1548–1557, Anchorage, Alaska, USA, Apr. 2001.

[4] J. Flinn and M. Satyanarayanan. Energy-aware adaptation for mobile applications. In *SOSP 99*, pages 48–63, Kiawah Island, SC, USA, Dec. 1999.

[5] P. Havinga. *Mobile Multimedia Systems*. PhD dissertation, University of Twente, Department of Computer Science, Feb. 2000.

[6] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *MobiCOM 99*, pages 174–185, Seattle, WA, USA, Aug. 1999.

[7] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *ASPLOS 00*, pages 93–104, Cambridge, MA, USA, Nov. 2000.

[8] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *MobiCOM 2000*, pages 56–67, Boston, MA, USA, Aug. 2000.

[9] R. Kravets, C. Carter, and L. Magalhaes. A cooperative approach to user mobility. *ACM Computer Communications Review*, 31, 2001.

[10] R. Kravets and P. Krishnan. Power management techniques for mobile communication. In *MobiCOM 98*, pages 157–168, Denver, Colorado, USA, Sept. 1998.

[11] A. Lebeck, X. Fan, H. Zeng, and S. Ellis. Power aware page allocation. In *ASPLOS*, pages 105–116, Cambridge, MA, USA, Nov. 2000.

[12] C. Miner. Pushing functionality into even smaller devices. *Communications of the ACM*, 44(3):72–73, 2001.

[13] M. Papadopouli and H. Schulzrine. Network connection sharing in an ad hoc wireless network among collaborative hosts. In *NOSSDAV 99*, Bell Labs, New Jersey, USA, June 1999.

[14] M. Papadopouli and H. Schulzrinne. Seven degress of separation in mobile ad hoc networks. In *Globecom*, San Francisco, CA, USA, Nov. 2000.

[15] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Sigcomm 2001*, San Diego, CA, USA, Aug. 2001.

[16] M. Roussopoulos, P. Maniatis, E. Swierk, K. Lai, G. Appenzeller, and M. Baker. Person-level routing in the mobile people architecture. In *USITS*, Boulder, Colorado, USA, Oct. 1999.

[17] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Middeware*, Heidelberg, Germany, Nov. 2001.

[18] A. Rudenko, P. Reiher, G. Popek, and G. Kuenning. Saving portable computer battery power through remote process execution. *Mobile Computing and Communications Review*, 2(1):19–26, 1998.

[19] M. Satyanrayanan. Pervasive computing: Vision and challenges. *IEEE Personal Communications*, 8(4):10–17, Aug. 2001.

[20] M. Stemm and R. Katz. Measuring and reducing energy consumption of network interfaces in hand-held devices. *IEICE Transactions on Communications*, Aug. 1997.

[21] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrisnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Sigcomm 2001*, San Diego, CA, USA, Aug. 2001.

[22] E. Swierk, E. Kiciman, N. Willams, T. Fukushima, H. Yoshida, V. Laviano, and M. Baker. The roma personal metadata service. In *IEEE Workshop on Mobile Computing Systems and Applications*, Dec. 2000.

[23] G. Zipf. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, Reading, MA, 1949.