

# An Energy-Efficient Middleware for Supporting Multimedia Services in Mobile Grid Environments

Yun Huang, Shivajit Mohapatra and Nalini Venkatasubramanian  
School of Information & Computer Science,  
University of California, Irvine, CA 92697-3425, USA  
{yunh, mopy, nalini}@ics.uci.edu

## ABSTRACT

In this paper, we present techniques for exploiting intermittently available resources in grid infrastructures to support QoS-based multimedia applications on mobile devices. Specifically, we integrate power aware admission control, grid resource discovery, dynamic load-balancing and energy adaptation techniques to enable power deficient devices such as PDAs to run distributed multimedia applications. Our integrated solution adapts to dynamic changes in device energy consumption and unpredictable grid resource availabilities without compromising application Quality of Service (QoS). Our simulation results indicate that our power-aware grid-based approaches, not only improve QoS of mobile multimedia services, but also efficiently load-balance the grid resources.

**Keywords:** Mobile Grids, Middleware optimizations for Grid, Energy Awareness

## 1. INTRODUCTION

Providing support for multimedia applications on low-power mobile devices remains a significant research challenge. This is primarily due to two reasons: (i) Portable mobile devices have modest sizes and weights, and therefore inadequate resources - low CPU processing power, display capabilities, limited memory and battery lifetimes as compared to desktop and laptop systems. (ii) On the other hand, multimedia applications tend to have distinctive QoS and processing requirements which make them extremely resource-intensive. This innate conflict introduces key research challenges in the design of multimedia applications, distributed adaptation algorithms and device-level power optimizations.

The grid computing environment [1] provides an ideal setting for applying proxy based techniques to aid and improve power/performance of low-power mobile devices. Conventionally, a grid consists of an aggregation of networked computers forming a large-scale distributed system that can be exploited to solve computational and data intensive problems. However, in [4], the authors motivate the use of low-power devices in the grid and present the challenges involved in integrating mobile devices into a grid and harnessing their resources. They suggest the use of a proxy based clustered architecture to support mobile nodes within a grid. While traditional grid based research has focused on making grids more flexible, secure, exploring effective resource sharing techniques and addressing other issues like scheduling, co-allocation and accounting, we investigate how grid resources can be effectively utilized to support multimedia streaming to mobile low-power devices. More specifically, we try to harness the idle resources of the computational grid (to act as “proxies”) to provide “energy-aware” services to power constrained mobile hosts, while ensuring that services are not interrupted either due to intermittent availability or overloading of grid resources.

However, making the grid “power-aware” poses several challenges. Firstly, grid resources are intermittently available; therefore scheduling policies must take availability of grid resources into account. Secondly, effective resource allocation policies must be designed to address the performance-quality tradeoffs for the multimedia applications. Thirdly, the energy constraints of devices, limited capability of grid resources for handling multiple services and mobility of the devices necessitate the need for efficient and adaptive resource management policies. To overcome the aforementioned challenges, in this paper, we propose grid resource allocation techniques that exploit knowledge of device mobility patterns, energy information and available idle grid resources; we also develop dynamic adaptation algorithms to enhance the overall performance of mobile multimedia applications within a grid. Specifically, we: (1) propose a global power-aware request-admission algorithm; (2) integrate proxy-based strategies into the framework for adapting to dynamic residual energy changes of the low-power devices; (3) develop strategies to handle unprecedented variations in grid resources (e.g. grid machine failure). These techniques are integrated into our middleware based grid framework called PAGODA (Power-Aware Grid Optimizations for mobile Multimedia Applications). In the following sections, we will discuss the PAGODA framework and present our integrated solution with energy aware grid optimization techniques. Subsequently, we will present the details of our simulation model and experimental details.

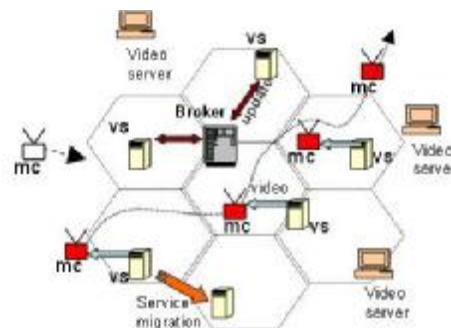
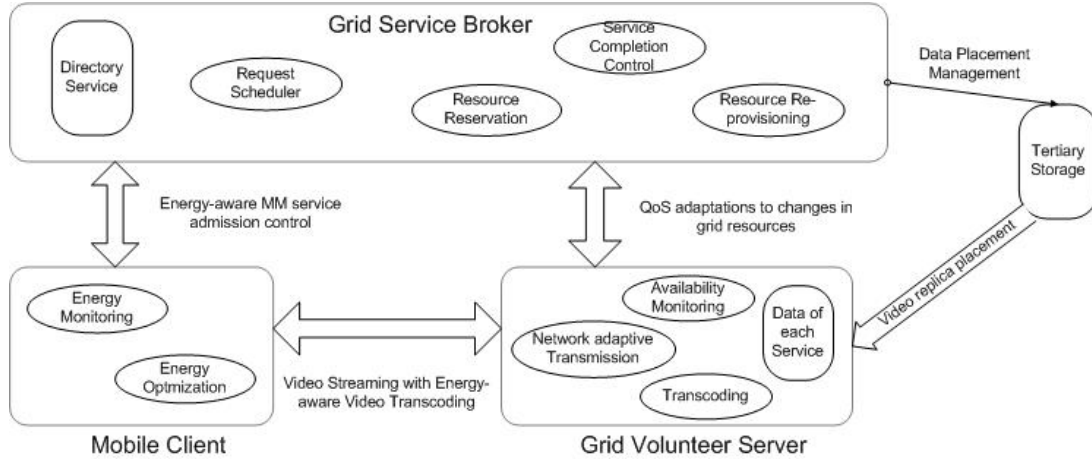


Fig 1. System Application Scenario

## 2. THE PAGODA FRAMEWORK

In this section, we introduce the PAGODA middleware framework for providing efficient multimedia streaming to mobile clients in grid based environments. Fig 1 shows the system application scenario<sup>1</sup>. A mobile client (MC) (e.g. PDA, laptop etc.) runs a streaming multimedia application (e.g. MPEG player) while traversing a series

<sup>1</sup> The mobile hosts connect to the infrastructure using a locally available wireless network. We do not consider ad hoc wireless networks, and we do not address the issues of wireless channel allocation (typically a MAC layer issue).



**Fig.2. The PAGODA Middleware Framework: for the grid-based mobile service system with QoS-based grid resource provisioning and energy-aware adaptive control on client's device (Volunteer Server is a grid resource provider).**

of cells (in a cellular network), and it can receive video streams from various grid resource providers (i.e. VS in Fig.1). We define a *Grid Volunteer Server (VS)* as a machine that participates in the grid by supplying idle resources when they are not being used, i.e. VSs are intermittently available. A VS in our case can be a wired workstation, server, cluster etc., which provides high capacity storage to store the multimedia data, CPU for multimedia transcoding, decompression and/or buffer memory. Additionally, the VS can be used to perform adaptive network transmission [9] that facilitates effective dynamic power management of device NICs (network interface cards).

In Fig.2, we show the various middleware services executing on the three system entities (i.e. Broker, the VS and the MC) with the arrows representing communications between the components. A *Grid Service Broker* forms a key component of the system and is assumed to be a high end server. The *Broker* is responsible for global adaptations in PAGODA. It performs the power aware admission control (i.e. it decides whether to accept or reject the request based on resource availability at the VS, the video quality requested and the residual battery capacity of the device) for incoming requests. It also determines the grid entities (VSs) that would serve the request over its entire duration and handles VS failures and service migrations within the grid.

The high-level algorithms executing on the three entities are shown

in Fig 3. For instance, the functionality of the Broker is implemented by various middleware services (in Fig 2), and include a Request Scheduler, a Resource Reservation service, functionality for dynamic Resource Re-provisioning and a Service Completion Control module to deal with VS failures. The *Directory Service (DS)* module in the *Broker* provides global state information about clients and resource availabilities at the VSs. Each VS downloads video replicas from a tertiary storage and performs dynamic video transcoding with adaptive network transmission. Similarly the low-power device has middleware services for monitoring residual battery energy and performing power optimizations (e.g. DVS).

When a new request is made by a mobile client (MC), the *Broker* executes the energy-aware admission algorithm (discussed in Sec. 4) based on the residual energy of the MC with the resource and network information retrieved from *Directory Service (DS)*. If no scheduling solution exists, the *Broker* rejects the request; otherwise it identifies a set of VSs that can serve the request at various times and reserves the necessary resources on them. The selected VSs replicate video segments from a tertiary storage accordingly; the DS is updated to reflect the new allocation information subsequently. Once a request is accepted, the framework performs several dynamic adaptations (discussed in Sec. 4) based on the local device/VS level changes (e.g. device energy changes, VS resource availability changes) and global system wide changes (e.g. VS failures) to

```

Grid Service Broker () {
  WHILE (TRUE)
  SWITCH ( Incoming Event)
  CASE NEW_REQUEST:
    Energy-aware_Admission_Control (Rid);
  CASE REQUEST_INTERRUPTED:
    Release_Preassigned_VS (Rid);
  CASE VS_RESOURCE_UPDATION:
    IF (RESOURCE_REDUCED)
      Resource_Re-Provisioning (VSid);
    ELSE
      IF (RESOURCE_INCREASED)
        Directory_Service_Updation(VSid);
  CASE VS_UNAVAILABLE :
    Service_Completion_Control(VSid);
}

Mobile Client () {
  Send request R to Broker with QH and QL and Er;
  IF (request accepted) {
    WHILE (in this video streaming service) {
      Receive video streaming from VS;
      Send periodic residual energy information to VS;
      Update location periodically to directory service;
    }
    IF (running out of battery)
      Shut down this service, and notify current VS.
  }
}

Grid Volunteer Server () {
  WHILE (TRUE)
  Receive new schedule from the Broker;
  VS_Local_Adaptations ();
}

```

**Fig. 3 High Level Algorithms executed at the (a) Broker, (b) Mobile Client (MC) and (c) Grid Volunteer Server (VS)**

maintain the highest possible QoS for each request. Note that the *Broker* adapts only to global changes that affect the entire system. For example, when it detects a *VS* failure, it reschedules the services pre-assigned to that *VS* onto other *VS*s. The *VS*s locally adapt video streams to improve QoS based on their resource availability (e.g. network transmission bandwidth) and current residual battery on the mobile devices.

In this paper, we assume network connectivity at all times and the middleware framework handles the protocol interactions and connection handoffs resulting out of tearing and re-establishment of connections. We are currently addressing these issues in another work and focus primarily on the energy and load balancing issues in this paper.

### 3. ENERGY-AWARE ADMISSION CONTROL

Our adaptive strategies are based on the fact that streaming lower quality video (e.g. through proxy-based transcoding) to power deficient mobile clients results in sending lighter traffic over the network and less computation for decoding video frames and therefore less energy consumption at the mobile device. However, the lower quality directly affects user perception of video (QoS), so it is important to understand the notion of video quality for a handheld device and its implications on power consumption. Before presenting our energy-aware grid-based solutions, we introduce an *E-Q* (*Energy/Quality*) matrix (Table 1) for handheld computers (Compaq iPaq 3650), which leverage on work of [9] to identify video quality parameters (a combination of *bit rate*, *frame rate* and *video resolution*) that produces a user perceptible change in video quality and a noticeable shift in power consumption for handheld computers.

Table 1. E-Q (Energy/Quality) matrix

QUALITY	Video transformation parameters	Avg. Power Windows CE (w)	Avg. Power Linux (w)
Q8 (original)	SIF, 30fps, 650Kbps	4.42	6.07
Q7 (Excellent)	SIF, 25fps, 450Kbps	4.37	5.99
Q6 (Very Good)	SIF, 25fps, 350Kbps	4.31	5.86
Q5 (Good)	HSIF, 24fps, 350Kbps	4.24	5.81
Q4 (Fair)	HSIF, 24fps, 200Kbps	4.15	5.73
Q3 (Poor)	HSIF, 24fps, 150Kbps	4.06	5.63
Q2 (Bad)	QSIF, 20fps, 150Kbps	3.95	5.5
Q1 (Terrible)	QSIF, 20fps, 100kbps	3.88	5.38

Note that the values provided in Table 1 will be different for different devices as well as various models of the same device. However, as profiling is a one time effort, we assume that such tables will be available for all mobile devices within the system. Therefore, in the rest of this paper, we apply table 1 to perform our energy-aware *VS* based dynamic video transcoding in PAGODA. Basically, using the *E-Q* matrix, we map the video quality levels  $Q_i$  to a network transmission bandwidth value  $NBW_i$ ; and a power cost value, and vice versa, e.g.:

$$P(Q_7) = TABLE1(EXCELLENT) = 4.37W .$$

A video streaming request  $R < VID, T, Q_L, Q_H, E_R, itinerary(opt)>$  specifies the required video object by *VID*, whole service period *T*,

the lowest QoS level  $Q_L$  and highest QoS level  $Q_H$ , current residual energy  $E_R$ , and the mobile client's *itinerary* (optional). When the *Broker* receives a new request, it needs to decide whether to accept or reject the request. Specifically, it needs to check: whether the device has sufficient battery energy left to playback the entire duration of the requested video at the desired quality; whether it can assign a set of *VS*s to the request and ensure that the *VS*s have the necessary resources to serve the request.

The *Broker* runs the Energy-aware Admission Control Algorithm (EAC) algorithm (Fig 4) to firstly determine the highest QoS level ( $Q_h \geq Q_L$ ) that the residual energy ( $E_R$ ) on mobile client can support. If the residual energy on the device is insufficient even for the lowest quality required by the client, the *EAC* algorithm rejects the request; otherwise, it attempts to identify a set of appropriate *VS*s to schedule the request. Details of the scheduling algorithm, *Partition\_Service\_Period()* and *VS\_Allocation(Q\_h)* (Fig 4), have been addressed in our previous work [11], therefore will not be reiterated in this paper. Briefly, to ensure effective load balancing of the grid resources (*VS*s) and guaranteed QoS to the mobile device under current system conditions, *Partition\_Service\_Period()* is used to divide the whole service period into non-overlapping chunks (possibly of different sizes), which may apply knowledge of user mobility pattern (given *itinerary*). Subsequently, *VS\_Allocation(Q\_h)* maps each chunk to a suitable *VS* by choosing one *VS* that is lightly loaded and geographically close to the mobile client [11]. Eventually, if it is possible to find *VS*s for all chunks, there exists a scheduling solution; otherwise, the request is rejected.

```

EAC ( R < Obj, T, Q_L, Q_H, E_R > {
    Q_h = Rule-based_QoS_Mapping (E_R, T, Q_L, Q_H);
    // if no sufficient energy for Q_L, then Q_h = -1.
    IF (Q_h != -1) // energy at least sufficient for Q_L
        BOOLEAN found = TRUE;
        Partition_Service_Period();
        FOR each chunk
            IF (VS_Allocation(Q_h) == null)
                found = false;
                record the available VS with maximum
                resources for this service period;
            IF ( NOT found)
                NBW_m = MIN(NBW of available VSs);
            IF (NBW_m >= NBW.TABLE1(Q_L))
                found = TRUE;
                Q_h = TABLE1(NBW_m);
            IF (found)
                ACCEPT(R);
                reserve resources;
            ELSE
                REJECT(R) // insufficient VS resources
        ELSE
            REJECT(R) // insufficient device energy.
    }

Rule-based_QoS_Mapping (E_R, T, Q_L, Q_H) {
    P_h = E_R / T;
    IF (P_h < TABLE1(Q_L)) Q_h = -1;
    ELSE
        IF (P_h < TABLE1(Q_H)) Q_h = Table1(P_h);
        ELSE Q_h = Q_H
    RETURN Q_h;
}

```

Fig 4. Energy-aware Admission Control Algorithm (EAC)

## 4. DYNAMIC ADAPTATIONS IN PAGODA

Within a grid environment, dynamic and unpredictable system changes can occur frequently (e.g. residual energy variations on the mobile device, unexpected resource fluctuations on the *VS* and even *VS* failures), which affect the request QoS, the service completion guarantees or both. It is therefore imperative that our system adapt dynamically to accommodate these changes. In this section, we identify and elaborate on our adaptation strategies to deal with dynamic changes.

Within PAGODA, dynamic adaptations can happen either at the *VSs* or at the *Broker*. While the *VSs* can adapt to smaller local changes, larger changes that affect the entire system require the involvement of the *Broker*. The *VSs* typically adapt services to handle unpredictable changes in the residual energy of the device (e.g. due to starting/stopping of applications on device, energy optimizations on device) and sudden changes in their local resource availability. However, if a *VS* fails, the *Broker* has to reassign services previously assigned to the failed *VS* by re-executing the *EAC* algorithm for the victim requests. Note that since these requests have already been admitted, the *Broker* must process them with higher priority as compared to new incoming requests.

### 4.1 Local Adaptations at Grid *VSs*

Fig. 5 illustrates the high level algorithm for *VS* based local adaptations in PAGODA. Given the latest residual energy feedback from the device, the *VS* dynamically determines the video quality to be streamed to each device; meanwhile, it must be able to handle unpredictable variations in its local resources. These conditions have been explained as two separate cases in the algorithm. When the *VS* receives the latest residual energy ( $e_r$ ) value from the device, it determines the highest quality video that can be streamed to the device using function of *Rule-based\_QoS\_Mapping* ( $e_r, t, Q_L, Q_H$ ) (in Fig 3) based on  $e_r$ , the remaining time of the service, its local resource availability and a system specific adaptation policy. If the device does not have sufficient battery energy to support the entire duration of the current service, then the *VS* streams video at the lowest acceptable quality, and notifies the device about its depleted battery energy state. This implies that the device is consuming too much power (maybe due to other applications) and only *VS* based adaptations cannot complete the service. On the other hand, if we employ energy optimization techniques [9] on the *MC*, then our *VS* based adaptation will eventually result in an increase in the video stream quality.

The *VS* also needs to perform dynamic adaptations when its own resources reduce unpredictably (e.g. the owner of *VS* starts various applications). If the resource changes are small, the *VS* performs local adjustments to satisfy the QoS requirements of the current set of services. This might require downgrading the video quality of an existing subset of services. On the other hand, if there is a significant change in the resource availability at the *VS* affecting the completion of certain services, then a subset of services have to be migrated away from the *VS*. In this case, the *VS* informs the *Broker* for initiating the service migration (sec 4.2). However, given system resource limitations, services that may not be schedulable on other *VSs* result in service failures. After migrating a service, the *VS* re-adjusts the released resources to accommodate the remaining services in order to minimize the number of migrations. We limit (place an upper bound) the number of migrations of a single service by bounding the maximum number of migrations over the lifetime of the service.

```
VS Local Adaptations_0 {
  WHILE (TRUE)
    SWITCH (Event)
      CASE DEVICE_ENERGY_UPDATE:
        Perform video QoS adaptations based on the device
        residual energy and the available resources on the VS
        using Rule-based_QoS_Mapping ( $e_r$ ) in Fig 3.
      CASE VS_RESOURCE_CHANGE:
        IF (resources increase)
          Send resource availability update to the Broker;
        IF (small resource reduction)
          Downgrade local services;
        IF (large resource reduction)
          Request broker for migrating services;
        IF (broker unable to re-schedule services)
          Service failed;
          Release resources.
}
```

Fig 5. *VS* Location Adaptation Algorithm

### 4.2 Global Adaptations at Grid *Broker*

Due to the dynamic nature of the grid, *VS* availability itself can change unpredictably, which can result in service failures. To deal with this problem, the *Grid Service Broker* (in Fig 1) needs to reallocate other *VSs* to the interrupted requests so as to fulfill their remaining service. When a specific *VS* becomes unavailable, the *Broker* retrieves information from the *Directory Service* about requests that are scheduled on the failed *VS*, and triggers the re-scheduling process for each invalidated service.

In order to reduce service failures and minimize service recovery time, we need to determine the order in which to migrate the disrupted services onto which *VSs*. To minimize service recovery time, the *Broker* classifies invalidated services into two categories: (1) services that have been started and (2) services are not yet started. Services in first category receive higher rescheduling priority than that of the second, whose service rescheduling can be postponed with an acceptable delay. If requests cannot be rescheduled, the *Broker* downgrades a number of the disrupted services to accommodate them in the remaining *VSs*; if they still are not reschedule-able even after downgrading the service, the *Broker* notifies the client that service fails and releases pre-allocated resources for the disrupted services on the other *VSs*.

## 5. PERFORMANCE EVALUATION

In this section, we analyze the performance of PAGODA system with various scheduling policies and adaptation strategies under different system configurations.

**The Simulation Environment:** In our simulation, we model the system environment as a cellular system with ‘*c*’ cells and ‘*v*’ *VSs* distributed within the area. Each *VS* has ‘*s*’ gigabytes storage, and ‘*n*’ Mbps network transfer bandwidth. The CPU and memory resources of the *VSs* are assumed not to be the bottlenecks. We use a service *TimeMap* for *VSs* to keep the information of when and how long the *VS* is available during the span of a day, and we apply three *TimeMap* models (i.e. *Uniform*, *Random*, *Total availability*) proposed in [11]. Note that the *TimeMap* can be changed any time due to dynamic changes in *VS* availabilities. We characterize incoming multimedia requests from a *MC* by using a Zipfian distribution [5]. To characterize mobility of individual nodes, we use the incremental

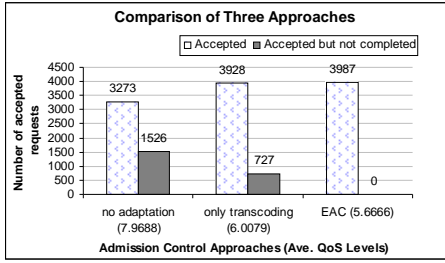


Fig 6 Comparison of 3 admission controls

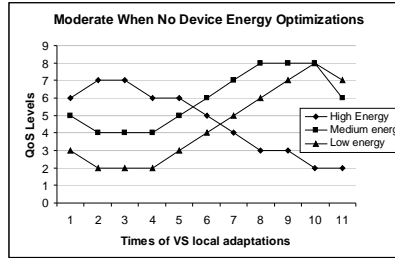
mobility model [6], where mobile hosts are distributed randomly and can move freely in a closed coverage area [11]. We set the duration of each video between 0 to ‘d’ hours. Each video replica requires ‘s<sub>i</sub>’ gigabyte disk storage, and network transmission bandwidth that ranges from 100 kbps to 1.3 Mbps. The shortest segment of a video object can run for ‘k’ minutes. In our simulation, we initially set  $c=100$ ,  $v=20$ ,  $s=100\text{GB}$ ,  $n=100$ ,  $d=2$ ,  $s_i=2\text{GB}$  and  $k=10$ , and the parameters are modified to simulate different configurations.

The device energy model is based on our experiments made on a typical handheld device, the Compaq iPaq 3600 running Windows CE. We use value 3.4 W as the power consumed when device CPU is idle, with super-bright back light and network interface card attached [9]. At the time of making the request, the average residual device energy on each device is generated using a random distribution. The Energy/Quality matrix presented in Table 1 is used to map video quality levels onto their corresponding power consumption levels. Additionally, applications are randomly started and stopped on the device to dynamically vary the energy consumption pattern with each application consuming an amount of energy varying randomly between 0.1W to 0.5W. We assume that the middleware on the device can determine the instantaneous residual energy  $E_R(t)$  of the device by making appropriate system calls.

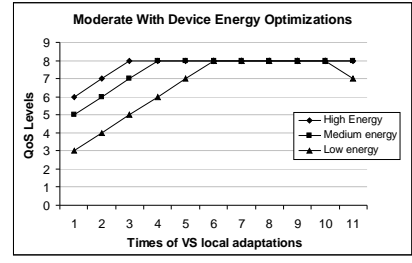
## 5.1 Experimental Results

The main metrics used to evaluate performance include: the system request acceptance ratio, the average QoS level of accepted services, and the number of incomplete services due to either insufficient energy on the device or reduced resources on VSs.

**Energy-aware Admission Control:** We evaluate the performance of the admission control using three different approaches: (1) no adaptation, (2) transcoding only and (3) the proposed EAC admission control algorithm. With approach 1, we stream the highest QoS level to the devices (i.e. multiple quality levels are not supported at the servers). With approach 2, we downgrade quality of service when the VS resources in the system are not sufficient for the highest QoS level. As seen from Fig. 6, approach 1 leads to lowest request acceptance ratio, but highest QoS provisioning for each service. However, as residual energy of the device is not considered, it also results in most number of incomplete services due to insufficient device energy. Approach 2 reduces the average QoS levels for requests, and therefore accepts more requests, while reducing the number of incomplete services due to insufficient device energy. Approach 3 (EAC) takes the residual energy of the device into account, while also performing quality transcoding; thus, it accepts the highest number of requests with all accepted services completed, assuming no other dynamic changes thereafter.



(a)



(b)

Fig 7 Impact of Device Energy Optimizations

**Local Adaptations at Grid VSs:** Table 4 illustrates the advantage of our VS based dynamic adaptations that account for resource availability changes and residual energy changes on the device. The first two columns indicate whether local VS adaptations are performed and whether power optimization is available at the devices. Clearly, the dynamic adaptations at the VS coupled with device level power optimization result in the best performance, with significantly reduced number of service failures and little difference in average QoS level.

Table 4. Effect of Local Adaptations at Grid VSs

VS Adapt	Power Opt.	Service Failure	Ave. QoS	QoS level Change
N	N	873	5.6664	0
Y	N	88	4.6995	- 0.9669
N	Y	0	5.6664	0
Y	Y	0	6.0246	+ 0.3582

**Impact of Device Energy Level:** Device level power optimizations (such as CPU voltage scaling [13], power mgmt. of network interfaces [9, 14]) continuously provide energy gains in the mobile devices. The local adaptation at the VS can leverage on these energy gains to boost the QoS of the service to the device. Fig 7 illustrates the adaptations for three randomly chosen requests (with high, medium, and low initial device energies) without (Fig 7a) and with (Fig 7b) device level energy optimizations. Clearly, with device level optimizations, a constant increase in the QoS is observed and a comparatively higher QoS is maintained for all requests. As we save residual energy on the device, we are able to increase the video qualities (Fig 7b).

## 5.2 Evaluating Performance under Changing Grid Conditions

The above simulation results indicate the effectiveness of our power-aware grid-based solution. We also study the system performance under changing grid conditions, e.g. varying *TimeMaps* and changing VS resources.

**Impact of Grid VS TimeMap:** Fig 8 shows how the dynamic availability of the VSs within the grid affects the number of rejected requests. Notice that if all VSs are available all the time, then there will be no rejections. However, when the VS availabilities change intermittently, the number of request rejections increase significantly over time. As expected, increasing the continuity of the *TimeMap* (e.g. from 3-hour to 6-hour continuous availability) improves the request acceptance ratio in this system without loss in the average

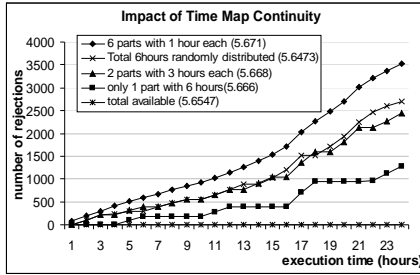


Fig 8. Impact of VS TimeMap continuity

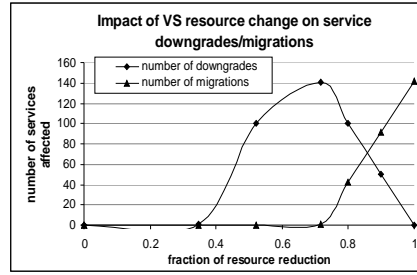


Fig 9. Impact of VS Resource Reduction

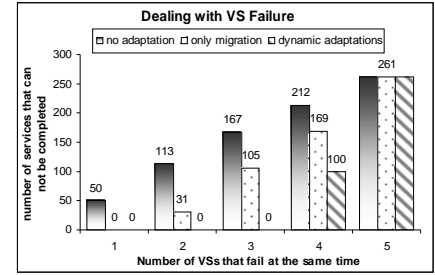


Fig 10. Dealing with VS Failures

QoS level (the number in the parenthesis marks for each condition represents the average QoS level of accepted services).

**Dealing with VS Resource Changes:** When local resources on the VS change unpredictably, the VS itself needs to adapt dynamically in order to maintain acceptable service QoS for all services executing on it. The VS can either downgrade local services to account for resource reduction or request the *Grid Broker* to migrate certain services. Various adaptation policies can be used to decide how many services to downgrade and the number of services to migrate. Fig 9 shows the performance of a simple policy that tries to minimize the number of migrations, which is more expensive than locally downgrading services. As seen in the figure, the VS can handle a resource reduction of as much as 71% by locally downgrading services. However, this significantly reduces the QoS of certain requests. Beyond that service migrations are necessary for service completions.

**Dealing with VS failures:** When dynamic changes to VS availability occur, a number of factors can affect the overall request completion ratios - the initial *TimeMaps* of each VS, the current load etc. Given the limited space, we only present the experiment results under a *Uniform availability TimeMap* and *Random* mobility pattern [11]. In the simulation, we choose the moment when a total of 5 VSs are available. We gradually increase the number of simultaneous VS failures and observe its impact on request completion ratios. As Fig 10 illustrates, we can significantly decrease the number of requests that fail to complete due to dynamic changes in VS availability by applying the adaptation strategy proposed in section 4.2. When the number of simultaneous VS failures increases to the extent that a large portion of the grid is unavailable, there are fewer overall resources available, causing increasing numbers of request completion failures. If all VSs become unavailable at the same time, the adaptation strategies cannot find any available VS to migrate the invalidated services; similar results are therefore observed with and without adaptations. When 4 VSs go down, the energy-aware admission scheduling and adaptive transcoding technique can still migrate large portion of services (177 migrations) successfully, and the average QoS level downgraded is only 1.12. Clearly, the enhanced solution for dealing with dynamic changes in VS availability brings much better performance to the system. Notice that the number of services that cannot be completed increases almost linearly in the graph, indicating that the system is “load-balanced” using our approach.

## 6. Related Work and Concluding Remarks

A tremendous amount of research efforts have been dedicated to achieving power savings in delivering multimedia services for mobile devices. Besides the research efforts we have cited in previous sections, battery power sensitive video processing and very-low-bit-rate video delivery techniques have been devised for streaming video

information in a heterogeneous mobile environment [12]. Algorithms that address the overlap between communications and compression have been proposed in [8]. Video segmentation [3], intelligent caching and buffer management techniques, and high speed wireless transmission have been extensively researched. In addition, resource management in the grid has been addressed by several grid projects such as Globus [1], Condor [10], AppLes [7], and Nimrod [2], etc. In this paper, we proposed the PAGODA framework for effectively utilizing available grid resources to improve global (grid-level) and local (device-level) system performance for mobile multimedia applications. We devised an energy-aware admission control algorithm that takes into account current device capabilities and grid resource availabilities to guarantee predictable services in mobile environments. We developed a family of QoS-aware adaptation policies that handle unforeseen variations in the grid resources. In future work, we plan to address issues like location awareness, secure communications and service reliability.

## REFERENCES

- [1] I. Foster, C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*, book 1998.
- [2] D. Abramson, J. Giddy, and L. Kotler, *High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid?* IPDPS 2000.
- [3] S. Chen, et. al, *Adaptive and lazy segmentation based proxy caching for streaming media delivery*, NOSSDAV, 2003.
- [4] T. Phan, L. Huang, C. Dulan, *Challenge: Integrating Mobile Wireless Devices Into the Computational Grid*, MobiCom, 2002.
- [5] A. Dan and D.Sitaram. *An online video placement policy based on bandwidth to space ration (bsr)*. SIGMOD, 1995
- [6] Z. Haas. *A new routing protocol for the reconfigurable wireless networks*, IEEE Universal Personal Communications, 1997.
- [7] F. Berman, R. Wolski, H. Casanova, et al. *Adaptive Computing on the Grid Using AppLeS*, HPCA, 1999.
- [8] E. Jeannot, B. Knutsson, M. Björkman, *Adaptive Online Data Compression*, 11<sup>th</sup> IEEE HPDC, 2002.
- [9] S. Mohapatra, R. Cornea, et.al., *Integrated Power Management for Video Streaming to Mobile Handheld Devices*, ACM MM, 2003
- [10] <http://www.cs.wisc.edu/condor/>
- [11] Y. Huang, Venkatasubramanian, *Supporting Mobile Multimedia Services with Intermittently Available Grid Resources*, HiPC 2003.
- [12] N. Yeadon, N. Davies, A. Friday, G. Blair, *Supporting Video in Heterogeneous Mobile Environments*, ACM SAC, 1998.
- [13] P. Pillai and K. G. Shin. *Real-time dynamic voltage scaling for low-power embedded operating systems*. SOSP 2001.
- [14] P. Shenoy and P. Radkov, *Proxy-assisted Power-friendly Streaming to Mobile Devices*, MMCN, 2003.