NAIST-IS-DD0461018

Doctoral Dissertation

QoS Adaptation Methods for Video Delivery in Pervasive Environments

Morihiko Tamai

February 1, 2007

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Morihiko Tamai

Thesis Committee:

Professor Minoru Ito	(Supervisor)
Professor Suguru Yamaguchi	(Co-supervisor)
Associate Professor Keiichi Yasumoto	(Co-supervisor)

QoS Adaptation Methods for Video Delivery in Pervasive Environments *

Morihiko Tamai

Abstract

In pervasive environments, mobile devices such as cell phones, PDAs and laptop PCs are connected through wired/wireless networks, and those devices are controlled based on user's context and preferences. In order to provide video streaming service in such pervasive environments, we need huge network and computational resources for encoding, delivering, decoding and drawing video contents. Thus, it is important how to achieve QoS which meets individual user requirements within various resource constraints such as network bandwidth, computational power on servers/clients, and battery amount. In this thesis, we propose application-level QoS adaptation mechanisms to provide satisfactory multimedia services to users in pervasive environments.

Contributions of this study are two folds. First, we propose an energy-aware video streaming system in which the video can be played back for the specified duration within the remaining battery amount. In the system, we execute a proxy server on an intermediate node in the network. The proxy server receives the video stream from the content server, transcodes it to the video with lower quality, and forwards it to user terminals. To predict required battery amount which enables playback for the specified duration and to control battery life, we have developed a power consumption model using parameters on playback quality, playback duration, battery amount and so on. The system also allows users to play back video segments with different qualities based on the importance

* Doctoral Dissertation, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0461018, February 1, 2007.

among video segments. Through experiments, we confirmed that our system is capable of controlling the playback duration within 6% of error, under dynamic QoS control.

Secondly, we propose a method to broadcast multimedia content for multiple mobile users with different quality requirements. In the method, we assume several proxies and wireless access points in the network. There are overlay links between these nodes, and certain amounts of bandwidths are reserved in advance. Each proxy is capable of executing multiple transcoding services and forwarding services. The original video sent from the server is transcoded into videos with various quality by these services, and delivered to user nodes with the required quality along appropriately determined service delivery paths. For this purpose, we propose an algorithm to construct the service delivery paths which minimize the weighted sum of computation power for transcoding and the bandwidth consumed on physical links on the overlay network. Through experiments with simulations, we confirmed that our algorithm can save up to 20% of resource amount consumed in a content delivery network compared with general algorithms.

Keywords:

QoS adaptation, energy-awareness, Content Delivery Network, transcoding

Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

Recent innovation and widespread of wireless network infrastructures has enabled pervasive environments where various types of portable computing devices (which we call *mobile terminals* or *mobile nodes*, hereafter) such as laptop PCs, PDAs, and cell phones are connected through wired/wireless networks, and those devices are controlled based on user's context and preferences. A lot of applications such as Web browsing, e-mail, on-line games, and navigation system are available for these mobile terminals in pervasive environments. Among them, video streaming is one of the most promising applications.

Streaming playback of video contents consume much more computational resources than other applications, due mainly to video decoding and wireless communication. Thus, in order to provide video streaming service in pervasive environments, it is important how to achieve QoS which meets individual user requirements within various resource constraints. Especially, because of the limitation of battery amount, there are demands for controlling battery life depending on user requirements, so that the battery is not exhausted during the video playback, or no more than the specific amount of battery is used by playing back the video. Therefore, we need a power consumption control mechanism in playing back a video for the specified duration within the specified battery amount.

From service provider's perspective, there is also a demand to reduce resource consumption in content delivery networks (CDNs) [1, 2, 3]. In order to achieve video delivery to multiple heterogeneous mobile terminals, we need, in addition to huge network resources, huge computational resources for transcoding video data to change video quality depending on resource constraints such as network bandwidth, computational power on servers/clients, screen size, and battery amount. Since available resources in CDN are limited, in order to support a large number of mobile terminals, we need a method to select delivery paths so that the total resources consumed will be as small as possible.

1.2 Our Approach

In this thesis, we report our research work to the aforementioned problems. This thesis consists of two parts.

First, in Chapter 2, we study a method for controlling battery life in video streaming on mobile terminals. In general, we can extend video playback duration by reducing video quality. However, it is difficult to estimate how long the battery lasts for each playback quality, since the battery consumption depends on not only video playback but also other factors such as OS, LCD, and so on [4]. Besides it, these factors are device dependent. Moreover, when the remaining battery amount is small and the playback quality is reduced over the whole playback duration, users may get frustrated. In order to mitigate such a situation, some fragments of a video important for a user should be played back with higher quality than others. Also, to each fragment, a user should be able to specify playback preferences such as balance of motion speed and vividness.

Power consumption when playing back streaming video on mobile terminals via wireless LAN consists mainly of (i) decoding and drawing video frames and (ii) packet transmission on wireless LAN. For (i), we can use a transcoding technique. For example, [5] proposes a power saving technique which reduces the total data size of the video by selectively dropping I, P and B frames when streaming MPEG videos. For (ii), there are several approaches. [6] proposes a power saving technique by regulating power for radio wave output in IEEE 802.11b wireless LAN. [7] proposes another power saving technique which reduces power consumption by shortening time for communication using data compression. These existing researches focus mainly on power saving techniques, but do not provide battery control depending on user requirements such as the playback duration within the battery amount.

In this thesis, we propose a method for energy-aware video streaming and playback which consists of (1) power-saving techniques for streaming video playback in mobile terminals on the wireless LAN, (2) a technique to estimate the suitable parameter values (such as picture size, frame rate and bitrate) which enable playback for the specified duration within the remaining battery amount, and (3) a QoS control mechanism which enables playback of multiple video segments with different playback quality based on the user's preference.

For (1), we introduce two techniques: (i) reduction of video quality with a transcoder; and (ii) periodic bulk transfer of the video data on the wireless LAN. For (i), we execute a proxy on an intermediate node (or on the content server) in the network so that the proxy receives the stream and forwards the transcoded stream to the user terminal. For (ii), the user terminal receives each fragment of the stream data at transmission rate as high as possible, stores the data in the local buffer, and stops supplying power to its WNIC (wireless LAN I/F card) until the buffer becomes empty. For (2), we have developed a power consumption model for the user terminal, in which the parameter values can automatically be decided from the desired playback duration, the remaining battery amount, and device specific information. For (3), we have developed an algorithm to distribute the battery amount among multiple video segments based on each user's preference. When a user specifies relative importance among video segments and preferred video property (proportion among picture size, frame rate and bitrate) for each segment, our algorithm determines the playback quality of each video segment so that the video playback can last for the specified duration within the remaining battery amount. Through experiments, we confirmed that our system is capable of controlling the playback duration within 6% of error under dynamic control of QoS parameters, and that the proposed algorithm can be applied to various mobile terminals by obtaining several device specific constants.

Next, in Chapter 3, we propose a method to broadcast multimedia content for multiple mobile users with different quality requirements. There are wide variety of screen size, computation power, battery amount, maximum network bandwidth in mobile terminals. Therefore, in order to broadcast video to multiple mobile terminals simultaneously, the following criteria should be considered and realized: (i) depending on constraints such as screen size, processing power, remaining battery amount, and possible transmission rate, each mobile terminal should be able to decide an appropriate quality of video to receive; (ii) contents provider should deliver video data of the requested quality. Letting a server deliver video to multiple mobile terminals by simultaneous unicast streams consumes a lot of computation and network resources. So, (iii) it would also be very important to save the resource amount consumed in a CDN so that the amounts of them are minimized.

There are many research efforts aiming at efficient broadcast of a video to multiple mobile terminals with different quality requirements. In the most promising technique (e.g., [8]), video data is encoded as a base layer and several extended layers using a hierarchical encoding technique such as MPEG-4 FGS [9] and those layers are broadcasted as separate multicast streams so that each mobile terminal can receive the base layer and a part of extended layers within its available bandwidth to playback video with the quality corresponding to the bandwidth. In this technique, however, extra memory is required for buffering all of receiving layers, and more computation power than decoding a single layered video is needed. Also, this technique has some drawbacks: the difference between the required quality and the received quality is large when there are only a small number of layers. Furthermore, it can only convert bitrate of video. Picture size and frame rate are fixed to the original value.

In this thesis, we propose a new service composition based method for efficient delivery of a video to multiple mobile terminals satisfying the above criteria from (i) to (iii). To achieve the criterion (i), each mobile terminal decides an appropriate quality of video depending on constraints such as screen size, available network bandwidth, and remaining battery amount using the aforementioned energy-aware video streaming technique. To achieve the criterion (ii), the proposed method utilizes transcoding service running on proxies for transcoding video to lower quality video. It also utilizes forwarding services on proxies to forward video to mobile terminals or to other proxies for transcoding the video to further low quality. To achieve the criterion (iii), we propose an algorithm to calculate service delivery paths among a server, proxies and mobile terminals (i.e., a set of delivery paths) on the overlay network as well as input/output video parameters (picture size, frame rate and bitrate) of each proxy so that the total resources consumed (both computation and network resources) will be as small as possible. Through experiments with simulations, we confirmed that our algorithm can save up to 20% of resource amount consumed in a CDN compared with simpler algorithms.

1.3 Thesis Outline

The rest of this thesis is organized as follows. In Chapter 2, we present an energy-aware video streaming system in which the video can be played back for the specified duration within the remaining battery amount. In Chapter 3, we propose a resource-efficient video multicast method to multiple mobile users with different quality requirements. Finally, in Chapter 4, we conclude this thesis.

2. Energy-Aware QoS adaptation for Video Streaming to Portable Computing Devices

2.1 Introduction

In this chapter, we propose a QoS adaptation method for streaming video playback to portable computing devices where playback quality of each video fragment is automatically adjusted from the remaining battery amount, desirable playback duration and the user's preference to each fragment.

In order to save power for streaming video playback, we introduce two techniques: (i) reduction of video quality with a transcoder; and (ii) periodic bulk transfer of the video data on the wireless LAN. For (i), we execute a proxy on an intermediate node in the network so that the proxy receives the stream and forwards the transcoded stream to the mobile terminal. For (ii), the mobile terminal receives each fragment of the stream data at transmission rate as high as possible, stores the data in the local buffer, and stops supplying power to its WNIC until the buffer becomes empty.

In order to estimate the suitable parameter values which enable playback for the specified duration within the remaining battery amount, we have developed a power consumption model for the mobile terminal, in which the parameter values can automatically be decided from the desired playback duration, the remaining battery amount, and device specific information. We have also developed an algorithm to distribute the battery amount among multiple video segments based on each user's preference. When a user specifies relative importance among video segments and preferred video property (proportion among picture size, frame rate and bitrate) for each segment, our algorithm determines the playback quality of each video segment so that the video playback can last for the specified duration within the remaining battery amount.

The rest of the chapter is organized as follows: in Sect. 2.2, we present the power saving techniques in playback of streaming video and the power consumption model to derive parameter values to satisfy the playback duration and the battery amount. In Sect. 2.3, we introduce our energy-aware QoS control technique based on relative importance among video segments. In Sect. 2.4, we

present our implementation of proposed system. In Sect. 2.5, we show evaluation of the proposed system through experiments and analysis. In Sect. 2.6, we briefly survey related work. In Sect. 2.7, we discuss applicability of our system to real environments. Finally, we conclude the chapter in Sect. 2.8.

2.2 Method for Energy-aware Video Streaming and Playback

In this section, we present two power saving techniques for playback of streaming videos on portable computing devices, and then introduce the power consumption model to determine the parameter values which enable playback for the specified duration within the battery amount.

2.2.1 Power Saving Techniques

When playing back a streaming video on a portable computing device, the consumed power consists of (i) power consumed by wireless transmission of packets, (ii) power consumed by decoding and drawing video frames, and (iii) power consumed by other factors such as operating systems, LCD back-light and so on.

When playing back an MPEG-1 stream (288×216 pixels, 24 frames per second, 327 Kbps bitrate) on a PDA via IEEE 802.11b WLAN (the device names are shown in Table 3), the power consumed by (i) and (ii) was more than 78 % of the whole power. So, it will be effective to save the power for (i) and (ii).

Power saving for decoding/drawing video frames

Since power consumed by video playback depends on the picture size, the frame rate and the bitrate, it is possible to save power by reducing the values of those parameters. In the proposed system, we adopt to use a transcoder called the *transcoding proxy* which receives the video stream from the content server, converts the stream to the stream with the lower parameter values in real time and forwards the new stream to the mobile terminal. The transcoding proxy can be executed on any node (including the content server) reachable from the mobile terminal in the network.

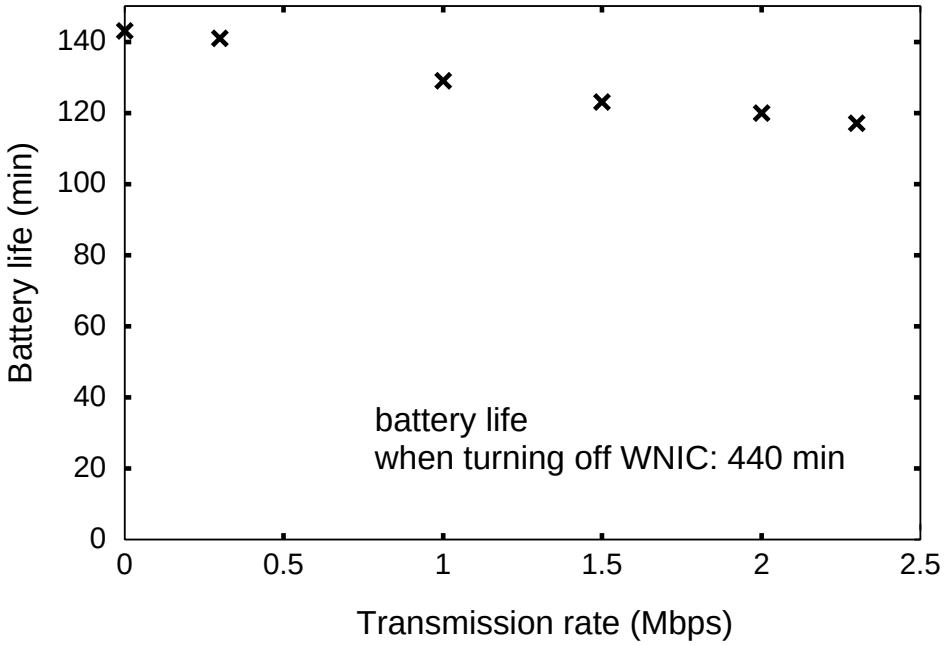

Figure 1. Battery life vs. transmission rate on WLAN

Power saving for wireless communication

We have investigated how the battery life changes depending on the transmission rate on the IEEE 802.11b WLAN. In the experiment, we used the PDA and the WNIC in Table 3. The result is shown in Fig. 1. Fig. 1 suggests us that power consumed by wireless communication increases in proportional to the transmission rate, but the minimum power consumed by the WNIC is much larger than the difference by transmission rate. So, if we can shorten the time to supply power to the WNIC, we can save the power.

In general, when playing back a video stream whose bitrate is b, the mobile terminal should receive the stream at transmission rate b and its WNIC should be always turned on during the playback. From the result in Fig. 1, we adopt the following periodic bulk transfer to save the power: the mobile terminal (1) receives each fragment of the stream data at the maximum available bandwidth more than b, (2) stores it in the local buffer and (3) turns off the WNIC while playing back the data in the buffer until the buffer becomes empty. We call this

scheme the *buffered playback*.

For push type video sources which transmit video streams at their bitrates, say, b, the buffered playback cannot be applied as is even when the available bandwidth is much larger than b. So, we let the transcoding proxy to store the transcoded data in its buffer until the buffer is filled, and transmit the data at the available transmission rate (denoted by b_{max}) to the mobile terminal. In this case, playback of the video is delayed for $(M/b_1) - (M/b_{max})$ which means the time until the buffer with M bit is filled. If the user specifies the maximum tolerable delay, buffer size M can be determined.

Power Consumption Model

We should be able to estimate the parameter values satisfying the specified playback duration and the remaining battery amount, for any combination of portable computing devices and WNICs. So, we construct an equation which represents the relation among the above parameters and device specific constants for mobile terminals.

Let us denote the powers consumed by playing back video and receiving video data by P_v and P_N, respectively. Let S denote the power consumed by other factors (operating system, back-light, memory and so on). Here, S is the device specific constant. Thus, power P consumed by the mobile terminal is represented by the following equation.

$$P = S + P_v + P_N \tag{1}$$

Power consumed by video processing

Playback of MPEG-1 videos needs two operations: decoding and drawing picture frames. The power consumed by drawing each frame is proportional to the number of pixels. The decoding operation consists of Huffman decoding, dequantization and inverse discrete cosine transformation. The power consumed by Huffman decoding is proportional to the bitrate of the video, and the power consumed by other processes is proportional to the number of pixels. Accordingly, the total power P_v to play back a video is represented by the following equation.

$$P_v(r, f, b) = \alpha r f + \beta b \tag{2}$$

where r, f and b denote the picture size (number of pixels), the frame rate and the bitrate, respectively, and α and β are device specific constants. Finally, we obtain a power consumption model for playback of a video

$$E_0 = (S + \alpha r f + \beta b)T \tag{3}$$

where T and E_0 are the expected battery life and the remaining battery amount.

Here, the actual values of S, α and β can be calculated by measuring the battery lives when playing back n videos with different parameters $\{(r_1, f_1, b_1), \ldots, (r_n, f_n, b_n)\}$, and using multiple regression. Since we have 3 variables, we should use $n \geq 3$.

Power consumed by communication

From the result in Fig. 1, we assume that power P_N consumed by communication is represented by

$$P_N(b) = N + \gamma b \tag{4}$$

where γ and N are device specific constants. From equation (1), we can derive

$$E_0 = (S + N + \gamma b)T \tag{5}$$

representing a relation among battery life T, battery amount E_0 and transmission rate b when the mobile terminal receives the video stream at b (without decoding and drawing video frames). The values of N and γ can be calculated by measuring battery lives when data are received at different transmission rates $b_1, ..., b_m$, and using multiple regression.

Power consumed by buffered playback

Let b_{max} denote the maximum available bandwidth at the mobile terminal. Let T_{on} denote the time interval to fill up the buffer since the buffer is empty. During this time interval, the WNIC is turned on. Let T_{off} denote the time interval to consume all of the data in the buffer since the buffer is full. During this interval, the WNIC is turned off. Let τ_{on} and τ_{off} denote the time intervals to resume the WNIC and to suspend it respectively. Let τ denote $\tau_{on}+\tau_{off}$. τ is a device specific

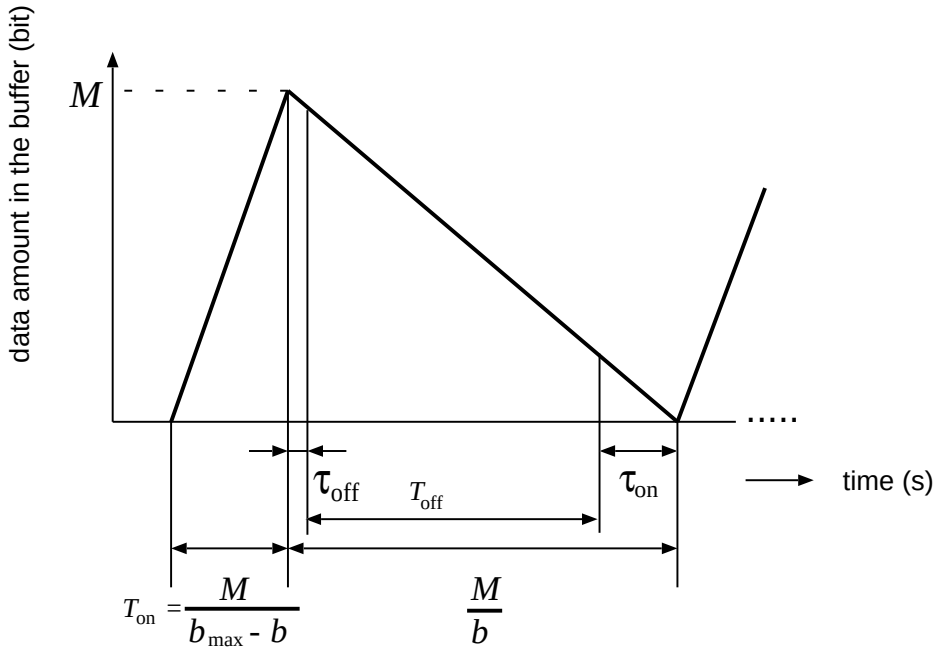

Figure 2. Time chart of buffered playback

constant and can be easily obtained (the actual value of τ in our environment was about 3sec).

Fig. 2 depicts the time chart of how the stream data is filled and consumed in the buffer when using the buffered playback, where M and b denote the buffer size and the bitrate of the video, respectively.

From Fig. 2, we obtain $T_{on} = \frac{M}{b_{max}-b}, \quad T_{off} = \frac{M}{b} - \tau$.

The power consumed by receiving the stream data is equal to $P_N(b_{max})$ while the WNIC is turned on, and 0 while turned off. So, the average power P_{buf} consumed by receiving the stream data (whose bitrate is b) is represented by

$$P_{buf}(b) = \frac{T_{on}}{T_{on} + T_{off} + \tau} P_N(b_{max}) = \frac{b}{b_{max}}(N + \gamma b_{max})$$

During time interval τ, the WNIC does not receive any data but consumes power to negotiate with the WLAN access point for authentication and address assignment by DHCP if necessary.

The average power P_{oh} consumed during time interval τ is represented by

$$P_{oh}(b) = \frac{\tau}{T_{on} + T_{off} + \tau} P_N(0) = \frac{b(b_{max} - b)\tau}{M b_{max}} N$$

Finally, we can derive a power consumption model when using the buffered playback.

$$E_0 = (S + \alpha r f + \beta b + P_{buf}(b) + P_{oh}(b))T \tag{6}$$

2.2.2 Algorithm for Deciding Parameter Values

Let (r_0, f_0, b_0) denote parameter values of the original video transmitted from the content server.

We assume that the desirable proportion among picture size r, frame rate f and bitrate b is given as follows.

$$r/r_0 : f/f_0 : b/b_0 = x : y : z \tag{7}$$

Equation (7) means that for example, if (x, y, z) are set to be $(1, 2, 1)$ and the picture size is reduced to 1/3 by the transcoding proxy, frame rate and bitrate are reduced to 2/3 and 1/3, respectively.

From equation (7), b and f can be represented by expression consisting of r and constants.

Consequently, when desirable playback duration T, remaining battery amount E_0 and the above proportion among r, f and b are given, we can calculate the values of r, f and b by solving equation (6). In our implementation, we have used Newton's Method to solve the equation.

Calculation of appropriate bitrate

When we specify the ratio among r, f and b by hand, the value of b may not be adequate due to quantization noise. We assume that the bitrate is linear to the picture size when the frame rate is fixed, and that the bitrate is linear to the frame rate when the picture size is fixed. From these assumptions, we can derive the following equation to obtain the appropriate bitrate from r and f.

$$b = c_0 r f + c_1 r + c_2 f + c_3, \tag{8}$$

where c_0, c_1, c_2 and c_3 are constants, and can be calculated using multiple regression.

We have prepared 64 videos with different parameter values which have similar balances of playback quality, and obtained the above constants using multiple regression. As a result, we got $c_0 = 7.9\text{e-}5$, $c_1 = 4.2\text{e-}4$, $c_2 = 13$, $c_3 = -16$.

2.3 Energy-aware QoS Control

In this section, we propose an algorithm to assign different playback quality among multiple video segments based on the user's preference, satisfying the specified playback duration within the remaining battery amount. Here, we assume that a video consists of multiple segments and those segments are classified into some predefined categories $C = \{c_1, ..., c_n\}$. Classification can be done manually using annotation tools like [10], or done automatically using tools like [11].

2.3.1 Specifying Importance among Categories

It is desirable for users to be able to specify what part of a video will be played back with higher quality. So, we allow users to specify relative importance among categories as *importance degrees*. Let p_i denote the importance degree specified to category c_i where p_i is an integer number such that $p_i \geq 1$.

The *playback property* of a video is decided by the balance of its picture size, frame rate and bitrate. In general, users may have different preferences for the playback property among categories. So, we allow users to specify a preference to the playback property of each category by the proportion among the picture size, the frame rate and the bitrate $r/r_0 : f/f_0 : b/b_0 = x : y : z$ as explained in Sect. 2.2.2. An integer number 1 or more than 1 can be specified to x, y and z. When we use equation (8), we need not specify b/b_0.

For example, suppose that a video of a soccer game consists of video segments classified into three categories {*shoot, play, other*}. A user may want to play back the video segments of category *shoot* at as high quality as possible, and the segments of category *play* at the medium quality, while reducing the playback quality of the segments of category *other*. The user may have the preference for playback property such that both the motion speed and the vividness are

Table 1. A sample user preference

category	importance degree	r/r_0	f/f_0	b/b_0
shoot	4	1	1	–
play	2	1	2	–
other	1	2	1	–

similarly important in category *shoot*, that the motion speed is more important in category *play*, and that the vividness is more important in category *other*. In such a case, the user gives the preference which is shown in Table 1.

2.3.2 Algorithm for Determining Playback Quality among Categories

For each category $c_i \in C$, the product of its importance degree p_i and playback duration T_i is called the *virtual playback time* of c_i. We denote it by $T'_i (= p_i T_i)$. Also, the total sum of the virtual time of all categories is denoted by $T' (= \sum_{c_i \in C} T'_i)$.

In our algorithm, we distribute the battery amount $E = E_0 - S \sum_{i=1}^{|C|} T_i$ among categories according to the proportion of each category's virtual time T'_i/T'. That is, $E_i (= ET'_i/T')$ is allocated for playback of each category c_i.

However, the above algorithm may result in over/under assignment of battery amount to some categories. We denote the properties of videos with the maximum quality and with the minimum quality by $(r_{max}, f_{max}, b_{max})$ and $(r_{min}, f_{min}, b_{min})$, respectively. Here, the maximum quality's video might be the video with satisfactory quality or the maximum video which the device can play back. The minimum quality's video can similarly be defined. So, we let our algorithm to adjust battery amounts in some categories so that restrictions of the maximum/minimum battery amount can be kept.

As explained in Sect. 2.2.2, battery amount E consumed by video playback is represented by equation (6) including parameters r, f, b and T. Consequently, if $E_i > (\alpha r_{max} f_{max} + \beta b_{max} + P_{buf}(b_{max}) + P_{oh}(b_{max}))T_i (\stackrel{def}{=} E_{max})$, E_i is too much for playback of c_i. Similarly, if $E_i < (\alpha r_{min} f_{min} + \beta b_{min} + P_{buf}(b_{min}) + P_{oh}(b_{min}))T_i (\stackrel{def}{=} E_{min})$, E_i is too small for playing back c_i. In those cases, we fix $E_i = E_{max}$ or $E_i = E_{min}$ and distribute remaining battery $E' (= E - E_i)$ among remaining

Table 2. Quality improvement when using bitrate extension algorithm

video			With bitrate extension (delay time=59sec)			Without bitrate extension		
segment	duration	priority	picture size	frame rate	bit rate	picture size	frame rate	bit rate
(No.)	(sec)		(pixel)	(fps)	(Kbps)	(pixel)	(fps)	(Kbps)
1	217	1	354×265	9.2	210	419×314	12.8	338.4
2	136	2	432×324	13.6	369.8	437×328	14	384
3	231	4	524×393	20.1	655.3	437×328	14	384
4	132	2	432×324	13.6	369.8	437×328	14	384
5	530	1	354×265	9.2	210	419×314	12.8	338.4
6	276	1	354×265	9.2	210	419×314	12.8	338.4
7	232	2	432×324	13.6	369.8	437×328	14	384
8	138	4	524×393	20.1	655.3	437×328	14	384
9	408	2	432×324	13.6	369.8	437×328	14	384

categories $C - \{c_i\}$. Consequently, we can obtain battery amount E_i for playback of category c_i as a constant value.

Using battery amount E_i, playback duration T_i, and the proportion $(r_i/r_0 : f_i/f_0 : b_i/b_0)$ and algorithm in Sect. 2.2.2, we can calculate parameter values r_i, f_i and b_i for each category c_i.

2.3.3 Bitrate Extension Algorithm for Low Bandwidth Environment

When available network bandwidth b_{max} at the mobile terminal is low (e.g., 128Kbps), playback quality cannot be improved better than this value even if its remaining battery amount is sufficient. To cope with this problem, we propose an algorithm to enable playback of some video segments with higher bitrates than b_{max}.

The proposed algorithm delays the start time of the video playback where the mobile terminal buffers enough amount of data before starting video playback.

We assume that we play back video segments $v_1, ...,$ and v_m in this order. Let p_i and T_i denote the importance degree and the playback duration of segment v_i. Let P_{min} denote the power consumed by the playback of video segments with the minimum importance degree. Let D denote the delay time to start the playback.

Then the following equation holds.

$$E_0 - S \cdot (D + \sum_{i=1}^{m} T_i) - D \cdot P_N(b_{max}) = \sum_{i=1}^{m} P_{min} \cdot p_i \cdot T_i \quad (9)$$

When battery amount assigned to segment v_i (represented by $P_{min} \cdot p_i \cdot T_i$) is determined, then we can obtain the corresponding r_i, f_i and b_i using the algorithm in Sect. 2.2.2. On the other hand, D can be calculated by

$$D = Max(\sum_{j=1}^{i-1} b_j \cdot T_j / b_{max} - \sum_{j=1}^{i-1} T_j), i = 2, ..., n+1 \quad (10)$$

Consequently, we can represent D as the function of P_{min}. From equation (9), we can calculate the value of P_{min} by using the Newton's Method.

We have applied the proposed algorithm to a video (640×480, 1150Kbps, 30fps, 2300sec) using E_0 as 50% of fully charged battery amount in the laptop PC environment (Toshiba SS S4/275PNHW) in Table 3, where we set b_{max} to be 384Kbps. The result is shown in Table 2. We see that bitrates much higher than b_{max} are assigned to video segments 3 and 8 whose importance degrees are 4, while the delay time is reasonable (59sec).

2.4 Implementation

We have implemented an energy aware video streaming system as shown in Fig. 3. In the system, the user specifies the location of the video, the playback duration and the preference including importance degrees to categories and the proportion $r/r_0 : f/f_0 : b/b_0$ to each category. The information is sent to the transcoding proxy with the terminal information including the remaining battery amount and so on. The transcoding proxy calculates the appropriate playback quality from the received information, receives the video stream from the content server, transcodes it to the stream with the playback quality obtained with algorithms in Sect. 2.2.2 and Sect. 2.3 and forwards the new stream to the mobile terminal.

We have implemented a movie player using Berkeley MPEG Player [12]. In the transcoding proxy, we have implemented the decision mechanism of playback quality in C, and have implemented the real-time transcoder for MPEG-1 streams using MJPEG Tools [13]. The transcoder first decodes an incoming MPEG-1

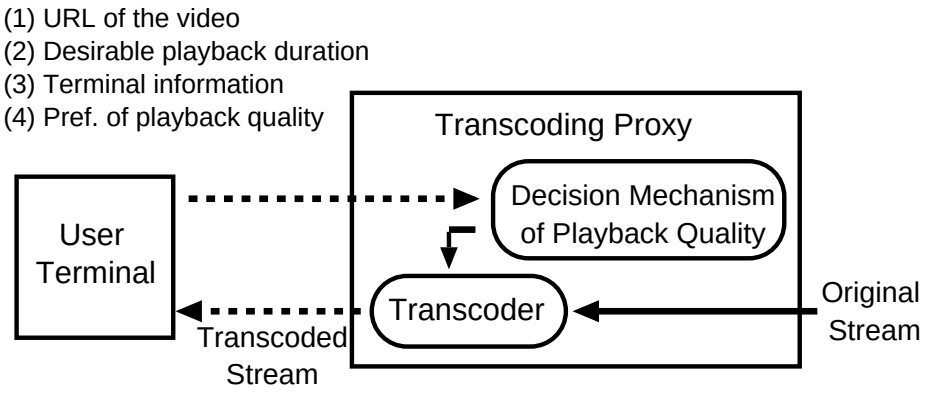

Figure 3. Energy aware video streaming system

stream to sequence of pictures in YUV format, and then resizes and drops YUV frames. Finally, YUV frames are re-encoded as an MPEG-1 stream. With this transcoder, each video can be transcoded to the one with any picture size, frame rate and bitrate.

It is desirable that transcoding can be done in real-time, and necessary computational resources are within realistic range. To confirm this, we measured processing time to transcode MPEG-1 streams using ordinary PC/AT compatible computer (Intel Pentium 4 2.40GHz, 1.0 GB RAM, Linux 2.4.18). Through the experiment, we have confirmed that our transcoder can convert videos with more than VGA size to any size, frame rate and bitrate in real-time.

2.5 Experimental Results

In order to evaluate the effectiveness of the proposed method, we have carried out several experiments using the environments in Table 3. In the experiments, we have used IEEE 802.11b WLAN. To restrain the power consumption due to rotation of HDD, we used `noflushd`[14] for the laptop PC environments.

Table 3. Experimental environments

device type; name; setting	WNIC type; name; setting	CPU; OS
PDA; Sharp Zaurus SL-C700; brightness small	CF; WN-B11/CF (I/O Data); low power mode off	XScale PXA250; Linux (Embedix)
laptop PC; IBM Thinkpad s30; brightness small	CF; GW-CF11H (Planex) low power mode off	PentiumIII 600MHz; Linux 2.4.27
laptop PC; Toshiba SS S4/275PNHW; brightness small	PCMCIA; GW-NS11S (Planex); low power mode off	PentiumIII 750MHz; Linux 2.4.24

2.5.1 Effectiveness of Power Saving Techniques and Power Consumption Model

In order to show effectiveness of our power saving techniques and validity of our power consumption model, we have measured the actual playback durations when using the buffered playback, and calculated errors from predicted durations.

In this experiment, we used the PDA and the laptop PC (IBM Thinkpad s30) environments. In order to obtain device specific constants S, α, β, we have measured actual battery lives using four videos with different quality for each environment as shown in Table 4. Also, for each environment, we have obtained constants γ and N by measuring actual battery lives when playing back a video with two different transmission rates as shown in Table 5. In the both environment, we have used the fully charged battery amount as E_0. The values of device specific constants are shown in table 6.

We used 2.3 Mbps and 1.8 Mbps as the average transmission rates for the PDA and the laptop PC, respectively, and used a buffer with 5 Mbyte. The experimental results are shown in Table 7. In Table 7, prediction error defined as $|t_e - t|/t$ is a difference between the predicted battery life t_e and the actual battery life t. Also, the actual battery lives in Table 7 are shown in Fig. 4 and 5 as graph.

Fig. 4 shows that the buffered playback could extend the battery life up to

Table 4. Videos used for measuring a value of S, α, β

Video parameter values (pixel, fps, kbps)	Battery life (min)
PDA	
(166 × 124, 24, 112)	309
(166 × 124, 24, 321)	293
(288 × 216, 24, 327)	216
(288 × 216, 8, 317)	294
laptop PC	
(320 × 240, 24, 299)	374
(320 × 240, 24, 813)	366
(560 × 420, 24, 818)	281
(560 × 420, 8, 788)	363

Table 5. Transmission rates used for measuring a value of N, γ

Transmission rate (Mbps)	Battery life (min)
PDA	
0.5	160
2.3	133
laptop PC	
0.5	395
2.0	362

Table 6. Device specific constant values for energy consumption model

Constant	PDA	laptop PC
S	$0.00254238E_0$	$0.00223922E_0$
α	$1.21931\text{e-}09E_0$	$2.15017\text{e-}10E_0$
β	$8.15911\text{e-}07E_0$	$1.29409\text{e-}07E_0$
γ	$0.000704887E_0$	$0.000153857E_0$
N	$0.00335518E_0$	$0.000215492E_0$
τ	3	3

about 2 times in the PDA environment. On the other hand, in the laptop PC environment, the effect of the buffered playback is not so much (Fig. 5). This is because the ratio of the power consumed by the WNIC to the whole consumed power is small in the laptop PC environment. When using the transcoding with the buffered playback, the battery lives could be extended up to about 2.8 times (PDA) and 1.6 times (laptop PC), respectively. The prediction errors were within 6% in both environments (Table 7).

2.5.2 Effectiveness of QoS Control

In this section, we assume that the laptop PC environment (Toshiba SS S4/275PNHW) is used and that 30% of the fully charged battery amount is used as E_0. An MPEG-1 video with 320×240, 29.97fps, 500Kbps and 1800 sec is used as the original video denoted by $v_0 = (r_0, f_0, b_0, T_0)$.

Playback quality in important categories

Using the algorithm in Sect. 2.3.2, we have investigated to what extent the playback quality of important categories is improved and the quality of the other categories is degraded.

We assume that video segments in v_0 are classified into two categories: important category c_1 and less-important category c_2. We assume that the preference to each category is $r/r_0 : f/f_0 = 1 : 1$ (b is automatically calculated using equation

Table 7. Actual and predicted battery life

Video parameter values (pixel, fps, kbps)	Battery life (min) actual/predicted (error %)	
	streaming playback	buffered playback
PDA		
(288 × 216, 24, 327)	120/122 (1.6 %)	189/185 (2.2 %)
(166 × 124, 24, 321)	142/143 (0.7 %)	242/240 (0.8 %)
(288 × 216, 8, 110)	149/150 (0.7 %)	289/285 (1.4 %)
(166 × 124, 8, 109)	163/160 (1.9 %)	340/323 (5.3 %)
laptop PC		
(560 × 420, 24, 818)	256/256 (0.0 %)	270/264 (2.3 %)
(320 × 240, 24, 813)	324/325 (0.3 %)	340/337 (0.9 %)
(560 × 420, 8, 286)	339/340 (0.3 %)	361/362 (0.2 %)
(320 × 240, 8, 294)	374/375 (0.3 %)	398/401 (0.7 %)

(8)).

Let R denote the ratio of playback duration T_1 of c_1 to total playback duration $T_1 + T_2$ (i.e., $R \stackrel{def}{=} T_1/(T_1 + T_2)$). Let p_1 and p_2 denote the importance degrees for c_1 and c_2, respectively. Let M denote the ratio of p_1 to p_2 (i.e., $M \stackrel{def}{=} p_1/p_2$). Using our algorithm in Sect. 2.3.2, we have calculated the playback quality in c_1 and c_2 by changing R from 0.1 to 0.3 by step 0.1 and M from 1 to 4 by step 0.5.

The resulting graphs are depicted in Fig. 6, where the horizontal axis and the vertical axis represent M and playback quality r/r_0, respectively. Here, r/r_0 becomes 0.58, if all categories have the same priorities, that is, $p_1 = p_2$ (baseline quality).

Fig. 6 shows that when R is small (e.g., 0.1) and we specify $M \geq 3$, the playback quality in important categories can be improved significantly by slightly reducing the playback quality of less-important categories.

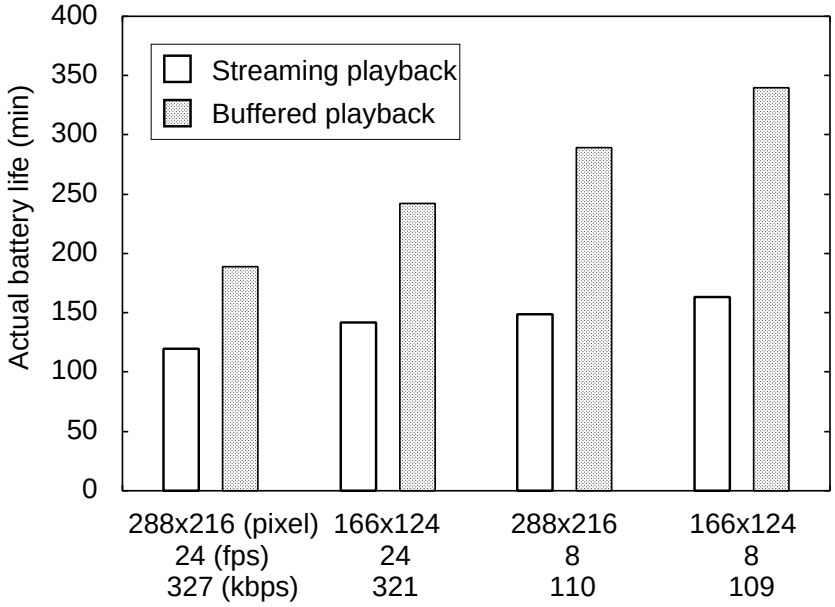

Figure 4. Actual battery life when using proposed method on the PDA

Prediction Errors When using QoS Control

To investigate the prediction errors of the actual playback duration within battery amount E_0 from its original playback duration 1800sec, we played back video v_0 at different playback quality calculated from four different preferences as shown in Table 8.

For *pref1*, *pref2*, *pref3* and *pref4*, actual playback durations until E_0 was exhausted were 1710sec, 1696sec, 1692sec, and 1727sec, respectively. The prediction errors are less than 6%.

2.6 Related Work

Our work is related to mainly on semantic transcoding and energy-aware video playback techniques.

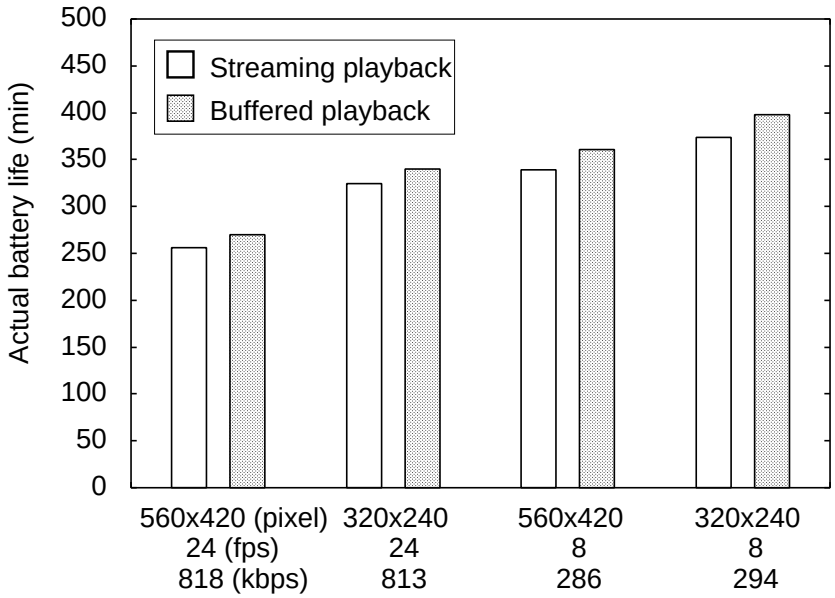

Figure 5. Actual battery life when using proposed method on the laptop PC

Semantic transcoding

In recent years, semantic transcoding techniques are paid much attention. Here, the user's satisfaction is improved by transcoding a video according to the contents and semantics of each fragment of a video [15, 16]. In previous transcoding techniques which simply reduce the picture size, objects in each picture frames becomes too small and difficult to identify. [15] copes with this problem by specifying the user's interesting area in the picture with the MPEG-21 DIA framework so that only the area is trimmed off and transcoded. In [16], a video in MPEG-4 format is divided into objects of several categories such as foreground objects and background objects. Here, playback qualities of important objects are maintained while qualities of other objects are degraded.

The objectives of these existing researches are to satisfy restrictions of portable devices with respect to picture size and available bandwidth. However, they do not treat restrictions on the battery amount.

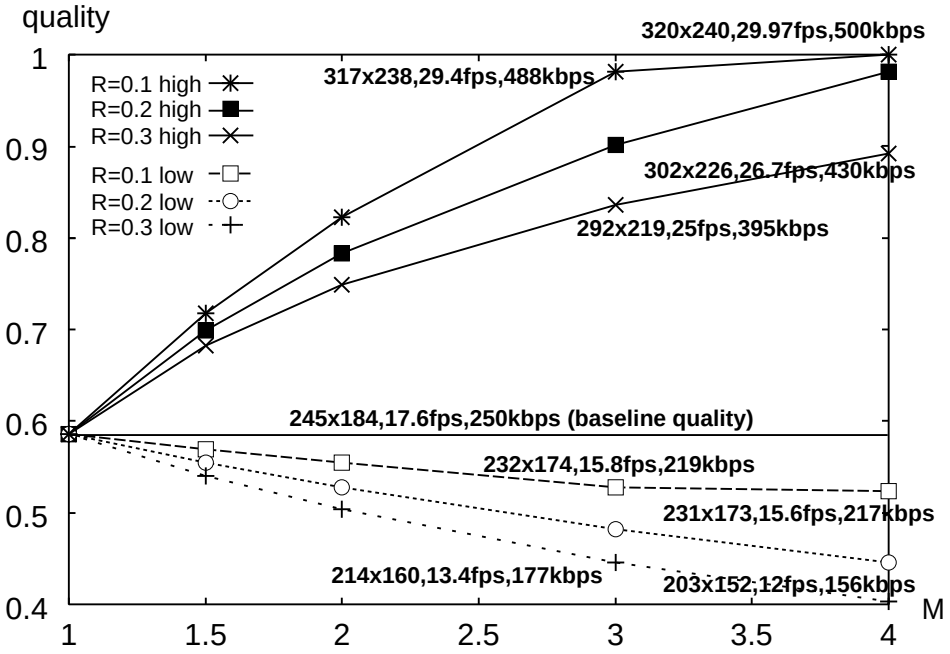

Figure 6. Quality improvement

Energy-aware video playback

There are many researches on dynamic voltage/frequency scaling (DVFS) based power saving techniques for video playback on mobile terminals [17, 18, 19, 20, 21]. DVFS reduces energy consumption by lowering the processor speed (voltage and frequency) dynamically at run time based on the system load. To determine the appropriate processor speed during video playback, these techniques predict exact amount of computation power needed to decode each frame by analyzing MPEG stream. In DVFS techniques, however, to achieve the expected performance, we need a processor which can scale to very precisely with respect to the voltage/frequency settings.

Recently, application-level adaptation techniques received increasing research attention [22, 23, 24]. In the technique proposed in [22], each application (such as a video player, a word processor, and a web browser) makes local power management decisions based on its processor demand and processor availability. They

Table 8. Preferences and playback qualities

	cat.	(T_i, p_i, f_i, r_i)	(r, f, b)
pref1	c_1	(678, 1, 1, 1)	$(245 \times 184, 17.60, 250K)$
	c_2	(662, 1, 1, 1)	$(245 \times 184, 17.60, 250K)$
	c_3	(460, 1, 1, 1)	$(245 \times 184, 17.60, 250K)$
pref2	c_1	(678, 1, 1, 1)	$(196 \times 147, 11.26, 143K)$
	c_2	(662, 2, 1, 1)	$(240 \times 180, 16.91, 238K)$
	c_3	(460, 4, 1, 1)	$(293 \times 219, 25.06, 395K)$
pref3	c_1	(678, 1, 1, 1)	$(196 \times 147, 11.26, 143K)$
	c_2	(662, 2, 1, 2)	$(292 \times 219, 12.47, 201K)$
	c_3	(460, 4, 2, 3)	$(320 \times 240, 21.61, 365K)$
pref4	c_1	(678, 1, 1, 1)	$(196 \times 147, 11.26, 143K)$
	c_2	(662, 2, 2, 1)	$(196 \times 147, 22.48, 288K)$
	c_3	(460, 4, 3, 2)	$(261 \times 196, 29.97, 439K)$

experimentally show that local decisions by individual applications can globally optimize system-wide energy usage and are better than choosing a single system-wide power setting for all applications.

In [23, 24], to reduce power consumed by WNIC, streaming data packets are transmitted as bursts, which allows WNIC to switch to energy-saving sleep mode between bursts. [24] explores theoretically and experimentally the impact of packet transmission interval and burst length, and proposes a mechanism that adjusts the burst length dynamically based on the network condition. [23] also proposes proxy-based power saving techniques which performs power-friendly video transcoding (i.e., reduction of picture size and frame rate) on the proxy to reduce the energy consumed by video decoding and drawing at mobile terminal. Although their power saving techniques are similar to our approach, they do not treat restrictions on remaining battery amount and desirable playback duration, and do not treat user's preferences such as relative importance among video segments and balance of picture size and frame rate.

2.7 Discussion

In this chapter, we have not assumed any particular hardware-level battery saving mechanisms for mobile terminals. Thus, our proposed method can be applicable to wide range of mobile terminals, which has no specific hardware, and be implemented entirely as application-level software without cost for additional hardware- and OS-level support.

Recently, processors such as Intel XScale [25] and Pentium M [26] have dynamic voltage and frequency scaling (DVFS) function. Using DVFS, we can reduce energy consumption by lowering the supply voltage when the CPU utilization is low. In such a case, energy consumption may further be reduced by using DVFS-based power saving methods with our proposed method. For this purpose, we need to modify our power consumption model to predict required power for decoding video data more accurately using existing power-aware video data processing algorithms proposed in [17, 18, 19] which determine the best processor speed by analyzing video data. However, since DVFS requires OS-level support, in order to use it, we may have to prepare and install special OS kernel to portable devices (e.g., Sharp Zaurus SL-C700 used in our experiments needs such a special kernel).

In Sect. 2.2.1, in order to reduce the power consumed by wireless communication, we proposed the buffered playback method which turns off the WNIC while playing back the video data in the buffer. However, this may incur an undesirable delay for reconnecting to the network (e.g., acquiring an address from DHCP server). In addition, the other applications (e.g., VoIP application which is waiting for the phone calls in the background process) may require sending/receiving control packets while turning off the WNIC. To address the problems, we can use low-power passive mode of WNIC which can save energy consumption significantly compared with the active mode while keeping the network connection [23, 24].

In Sect. 2.2.1, we assumed that the power consumed by the back-light of a mobile terminal is a constant value. However, in general, brightness of the back-light can be changed to several values (e.g., high, middle, and low). In order to take this into account, we can easily modify equation (1) so that different constant values S_i is used for i-th brightness of the back-light.

In this chapter, we have not considered the fluctuation of the available bandwidth at a mobile terminal. However, when applying the proposed method to real environments, fluctuation of the available bandwidth may occur and it causes mis-prediction of battery life. One approach to solve this problem is to rerun our algorithm periodically and adjust video quality to reflect changes in the available bandwidth.

2.8 Conclusions

In this chapter, we proposed an energy-aware video streaming system for streaming video playback on portable computing devices, based on the user requirement including the battery amount, the desirable playback duration and the relative importance among video segments. We have implemented the system and carried out some experiments using a PDA and laptop PCs on IEEE 802.11b WLAN.

Through experiments, we confirmed that (1) our system can control the playback duration within 6% error even when controlling QoS dynamically, and that (2) the proposed algorithm can be applied to various portable computing devices by obtaining several device specific constants. We believe that our model is accurate enough since some part of prediction errors is caused by the inaccuracy of the APM mechanism.

3. Resource-Aware Service Composition for Video Multicast to Heterogeneous Mobile Users

3.1 Introduction

In this chapter, we propose a new service composition based method for efficient delivery of a video to multiple mobile nodes. In the proposed method, we assume the following environments: (i) an overlay network connecting multiple proxies and a video server as shown in Fig. 7, is given as CDN. A fixed amount of bandwidth is assigned to each overlay link between proxies (or connecting to a server node) using existing network level QoS techniques such as DiffServ [27]; (ii) each proxy can execute multiple transcoder services and forwarding services within its available resources; (iii) To each proxy, at most one wireless access point (AP) can be attached; and (iv) each mobile node communicates with a proxy via the corresponding AP which is automatically and uniquely determined based on the current position of the mobile node.

In the proposed method, each mobile node decides an appropriate quality of video depending on constraints such as screen size, available network bandwidth, and remaining battery amount using our energy-aware video streaming technique proposed in Chapter 2. Here, appropriate quality (i.e., vector of picture size, frame rate, and bitrate) of each video segment is automatically determined from constraints of the mobile node and the user's requirement consisting of playback duration (e.g., time length of the video), relative importance among video segments and preferable ratio between picture size and frame rate for each segment. To deliver video data of the requested quality, the proposed method utilizes transcoding service running on proxies for transcoding video to lower quality video. It also utilizes forwarding services on proxies to forward video to mobile nodes or to other proxies for transcoding the video to further low quality. To save the resource amount consumed in a CDN, we propose an algorithm to calculate service delivery paths among a server, proxies and mobile nodes (i.e., a set of delivery paths) on the overlay network as well as input/output video parameters (picture size, frame rate and bitrate) of each proxy so that the total resources consumed (both computation and network resources) will be as small as possible.

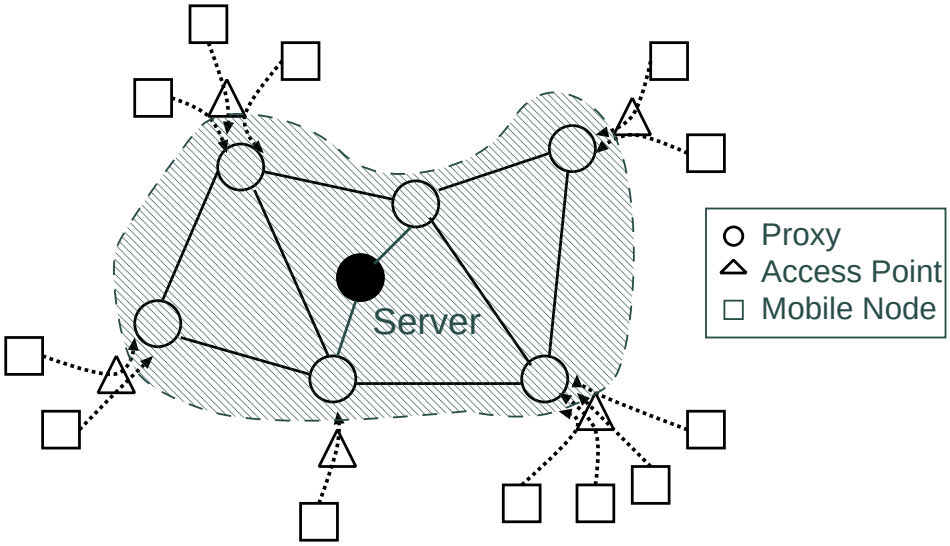

Figure 7. Network environment

The rest of the chapter is organized as follows: in Sect. 3.2, we describe the target environments and formally define the problem. In Sect. 3.3, we explain how the proposed system works on the target environments. In Sect. 3.4, we propose algorithms to calculate efficient service delivery paths for video multicast to multiple mobile nodes. Sect. 3.5 shows the experimental results. In Sect. 3.6, we present related work. In Sect. 3.7, we discuss some open problems. Finally, we conclude the chapter in Sect. 3.8.

3.2 Problem Definition

In this section, we describe the target environments and assumptions, and then formally define the problem to deliver video to multiple heterogeneous mobile nodes using the service composition technique.

3.2.1 Target Environments and Assumptions

We assume the existence of a content server, an overlay network, mobile nodes, service components, and proxies as follows:

1. Content server: A server transmits a recorded or live video (original video) to other nodes. Mobile user's quality requests are lower than the quality of original video. Starting time of the broadcast is predetermined similarly to TV broadcast.

2. Overlay network: An overlay network consisting of a video server, multiple proxies, multiple wireless access points (called AP, hereafter) and multiple mobile nodes is given in advance (Fig. 7). Here, a certain amount of bandwidth is reserved for video delivery on each overlay link using network level QoS techniques such as DiffServ [27]. At most one AP is attached to each proxy (multiple APs attached to a proxy can be regarded as one AP whose transmission bandwidth is the sum of their bandwidths). Available bandwidth between each proxy and the corresponding AP is larger than the upstream bandwidth of the proxy. Available bandwidth between an AP and the corresponding mobile nodes is larger than the sum of the transmission bandwidths of mobile nodes. Therefore, these links will not be bottlenecks during video delivery.

3. Mobile node: There are multiple mobile nodes (e.g., laptops, PDAs, cell phones, etc) which have different screen sizes, computation powers, available transmission speeds, and so on. They can communicate with a proxy via corresponding AP only when its radio range covers location of the mobile node. The corresponding AP can be uniquely determined from the location of each mobile node, and each mobile node immediately notices that it moves into a radio range of another AP. Mobile nodes do not exchange messages directly.

4. Service: There are two kinds of service components: (i) transcoding service and (ii) forwarding service. The computation powers required to execute these services can be calculated depending on input/output quality of the video and the input/output bitrates of the video, respectively.

5. Proxy: Each proxy has the maximum computation resources (CPU power, memory amount, and so on). Within these capacities, each proxy can instantiate arbitrary number of service components. For the sake of simplic-

ity, we treat only the computation power (i.e., CPU usage) required for execution of transcoding services.

3.2.2 Formal Definition of Problem

In this section, first, we present the notation of parameters used in the rest of this chapter. Then, we formally define the problem.

Notation and definition

Overlay network

Let s, $P = \{p_1, p_2, ..., p_{np}\}$, and $U = \{u_1, ..., u_{nu}\}$ denote a server, the set of proxies, and the set of mobile nodes, respectively. If a mobile node $u \in U$ is in the radio range of an AP and can communicate with a proxy $p \in P$ through the AP, we regard that there is an overlay link between u and p and it is denoted by (u, p). Let W denote the set of overlay links connecting to mobile nodes. Note that W changes as mobile nodes move. Let F denote the set of overlay links between nodes of $\{s\} \cup P$. Let $V = P \cup U \cup \{s\}$ and $E = W \cup F$ denote the set of all nodes and the set of all overlay links, respectively. Then, an overlay network is represented as a graph $G = (V, E)$. We denote the maximum available computation resource of each proxy $p \in P$ by $c\_avail(p_i)$, and the maximum available bandwidth of each overlay link (p_i, p_j) by $b\_avail(p_i, p_j)$ where $p_i \in P$ and $p_j \in \{s\} \cup P$. We denote the physical hop count of (p_i, p_j) by $hop(p_i, p_j)$. As we stated before, we assume that the maximum available bandwidth of each link $w \in W$ is not limited. An example overlay network is shown in Fig. 8 (a).

Required resource to execute transcoding service

We assume that the quality of video depends only on picture size (number of pixels), frame rate and bitrate. We denote these parameters by $q.s, q.f$ and $q.b$, respectively, and hereafter we call the quality of video the *quality vector* denoted by $q = (q.s, q.f, q.b)$. We assume that the required computation power to transcode the video with quality vector q to the video with q' can be represented as the sum of the powers decoding the video (including the power for processing the decoded pictures) with quality vector q and encoding the video with q'. We also assume that the required powers for decoding and encoding video are proportional to the number of pixels processed per unit of time based on the result in [28].

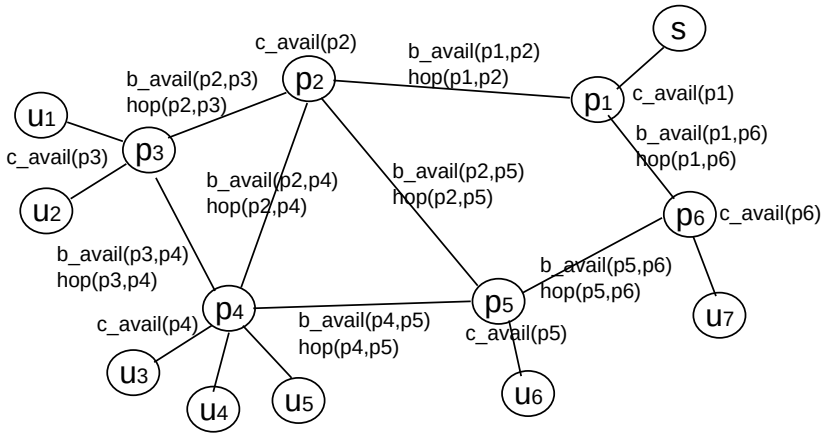

Figure 8. Example of overlay network topology

According to the above discussion, with some device specific constants τ_d and τ_e, the computation powers required for decoding and encoding video are represented by the following expressions.

$$r_{decode}(q) = \tau_d \times (q.s \times q.f) \tag{11}$$

$$r_{encode}(q') = \tau_e \times (q'.s \times q'.f) \tag{12}$$

Constraints on service paths

We want to calculate sequences of proxies with input/output quality vectors of video at each proxy to form so-called *service paths* from server s to all of mobile nodes U. On each service path, constraints on maximum available computation power at each proxy and the maximum available bandwidth on each overlay link must be satisfied. In Fig. 8 (b), we show an example of service paths for the overlay network of Fig. 8 (a), where mobile nodes $u_1, u_2, u_3, u_4, u_5, u_6$ and u_7 require video delivery with quality vectors $q_1, q_2, q_2, q_3, q_3, q_4$ and q_5, respectively. Here, $q_i.s \leq q_j.s \wedge q_i.f \leq q_j.f \wedge q_i.b \leq q_j.b$ if $i > j$.

For each node $v \in V$, we denote the set of quality vectors of videos which v receives, by $R(v)$. As special cases, we consider that $R(s) = \{q_{orig}\}$ and $R(u) = \{q_u\}$, where q_u is the required quality of mobile node $u \in U$. For example, in Fig. 8 (b), $R(p_5) = \{q_1, q_4\}$, $R(u_1) = \{q_1\}$ and $R(s) = \{q_0\}$.

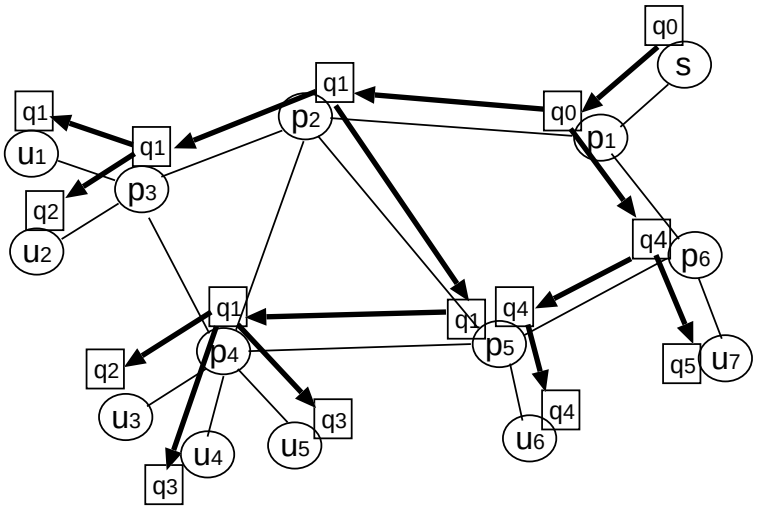

Figure 9. Example of service delivery paths

When $R(v) \neq \emptyset$ for a node $v \in P \cup U$, there should be v's parent node v' which transmits a video with quality vector $q \in R(v)$ to v, and v' should be receiving a video with q' such that $q.s \leq q'.s \wedge q.f \leq q'.f \wedge q.b \leq q'.b$. We call the relationship between v and v' as *forwarding relationship* and denote it by $(v', q') \rightarrow (v, q)$.

If $q' \neq q$, proxy v' must execute the transcoding service from video with q' to that with q. Otherwise, v' just executes the forwarding service to forward video to v without transcoding it. We denote the set of quality vectors input to the transcoding services executed at proxy p_i by $D(p_i) = \{q \mid (p_i, q) \rightarrow (v, q'),\ v \in P \cup U,\ q \neq q'\}$, and those output from the transcoding services at p_i by $E(p_i) = \{q' \mid (p_i, q) \rightarrow (v, q'),\ v \in P \cup U,\ q' \neq q\}$.

Required computation and bandwidth resources

We denote the required amount of computation power at proxy p_i and the required amount of bandwidth on overlay link (p_i, p_j) by $c\_cons(p_i)$ and $b\_cons(p_i, p_j)$, respectively. $c\_cons(p_i)$ and $b\_cons(p_i, p_j)$ are defined as follows.

$$c\_cons(p_i) = \sum_{q \in D(p_i)} r_{decode}(q) + \sum_{q \in E(p_i)} r_{encode}(q)$$

$$b\_cons(p_i, p_j) = \sum_{brate \in \{q'.b \mid (p_i,q) \to (p_j,q')\}} brate + \sum_{brate \in \{q'.b \mid (p_j,q) \to (p_i,q')\}} brate$$

Problem Definition

The problem is to calculate the set of quality vectors $R(p_i)$ of the videos which each proxy p_i receives, and the set of all forwarding relationships $(v', q') \to (v, q)\,(v' \in \{s\} \cup P, v \in P \cup U)$, satisfying the following constraints (13)–(16), when an overlay network $G = (V, E)$, the quality vector of the original video q_{orig}, and quality requirement q_u of each mobile node $u \in U$ are given.

$$\text{for each } (v', q') \to (v, q),\ q.s \leq q'.s \wedge q.f \leq q'.f \wedge q.b \leq q'.b \tag{13}$$

$$\text{for each } u_i \in U,\ \exists(s, p_{j1})\exists(p_{j1}, p_{j2})...\exists(p_{jk}, u_i)\ s.t.$$
$$(s, q_{orig}) \to (p_{j1}, q_1)\ \wedge\ (p_{j1}, q_1) \to (p_{j2}, q_2) \wedge \ldots \wedge (p_{jk}, q_{jk}) \to (u_i, q_{u_i}) \tag{14}$$

$$\text{for each } p_i \in P,\ c\_cons(p_i) \leq c\_avail(p_i) \tag{15}$$

$$\text{for each } (p_i, p_j) \in F,\ b\_cons(p_i, p_j) \leq b\_avail(p_i, p_j) \tag{16}$$

Constraint (13) represents that the parent node v' must receive the video with the higher quality vector in all quality parameters than v for transcoding and forwarding. Constraint (14) represents that there must be the sequence of overlay links connecting the server s and each mobile node $u \in U$ via a set of proxies. Constraints (15) and (16) represent that the consumed computation power at each proxy p_i and the consumed bandwidth on each overlay link (p_i, p_j) (or (s, p_i)) must not exceed the predetermined capacities.

In general, there may be multiple solutions which satisfy the above constraints. So, we use the following objective function to minimize the amount of consumed resources.

$$\text{Min } (\alpha \sum_{p \in P} c\_cons(p) + (1-\alpha) \sum_{\{p_i, p_j\} \in F} b\_cons(p_i, p_j) \times hops(p_i, p_j)) \qquad (17)$$

The first term of the objective function (17) represents the total sum of computation power consumed at proxies, and the second term does that of bandwidth consumed on overlay links between proxies considering their physical hop counts. Here, α is used to make which kinds of resources more expensive.

3.3 System Behavior

In this section, we describe how the whole proposed system works in detail. In Sect. 3.3.1, we will explain how each mobile node makes a request of its video quality taking into account of the amount of remaining battery. In Sect. 3.3.2, we explain a grouping method of quality requirements to reduce the number of different quality videos for pre-processing. And in Sect. 3.3.3, we will explain the communication protocol between nodes for video transfer.

3.3.1 Determining Required Quality Based on Battery Amount

Let bw_u, cp_u, ds_u and en_u denote available receiving bandwidth, available processing power, screen size and the amount of remaining battery for a mobile node $u \in U$, respectively. The request from a node u has to satisfy following restrictions.

$$\begin{aligned} q_u.b &\leq bw_u \\ q_u.s &\leq ds_u \\ \tau_d \times (q_u.s \times q_u.f) &\leq cp_u \end{aligned}$$

If the user of node u specifies time of duration T_u for watching a video, the amount of remaining battery has to be considered to decide video quality. We have already proposed a method to find a suitable video quality (a combination of screen size, frame rate and bitrate) for mobile node from time of duration T_u, the amount of remaining battery en_u and constants inherent to the model

of mobile node (e.g. power consumption of running OS, and so on) in [28]. If the video is not a live video and recorded beforehand, video segments in the video is known, and we assume that the contents provider assigns keywords to each segment by automatic labeling tools such as [10]. In this case, quality of each video segment can be changed according to relative importance of each video segment and preferred playback characteristics (faster framerate or higher resolution) specified by the user.

Hereafter, we describe how a user node decides video quality.

$C = \{c_1, ..., c_m\}$ denotes a set of categories (e.g., keywords) assigned to video segments. Each user specifies a relative importance p_i for each category in C. Here, p_i is an integer value larger than 0. The amount en_u of remaining battery is distributed among categories proportional to the product of total length T_i and specified importance p_i of category c_i.

That is, $\frac{en_u \cdot p_i \cdot T_i}{\sum_{j=1}^{m} p_j \cdot T_j}$ is the amount of battery used for playing back video segments which belong to a category c_i. As we described before, playback quality can be differentiated by specifying different playback characteristics, even if the amount of battery used for playback is same. In order to achieve this, the user specifies a ratio between screen size and framerate $q_u.s/q_{orig}.s : q_u.f/q_{orig}.f = x : y$ for each category. Here, x and y are integer numbers larger than 0. The quality of each video segment can be calculated using the method in [28]. We explain this method by an example soccer video consists of three categories $\{shoot, play, other\}$. Suppose a user specifies that he wants to see *shoot* scenes in higher quality, *play* scenes in medium quality, and *other* scenes in lower quality. Both screen size and framerate are important for *shoot* scenes, framerate is more important in *play* scenes, and screen resolution is more important in *other* scenes. In this case, he specifies as follows.

category	importance	$\frac{q_u.s}{q_{orig}.s}$	$\frac{q_u.f}{q_{orig}.f}$	Length
shoot	4	1	1	10min
play	2	1	2	35min
other	1	2	1	15min

Scenes in the category *shoot* are played back using

$\frac{en_u \cdot 4 \cdot 10}{4 \cdot 10 + 2 \cdot 35 + 1 \cdot 15} = \frac{8}{25} en_u$ of the remaining battery amount, and thus these scenes are played back in higher quality than others.

In the proposed method, video quality can be decided by the method explained above, for recorded video. On the other hand, when a live video is broadcasted, the method above cannot be used since categories of the video segments and their total lengths cannot be known beforehand. In this case, each user specifies a quality from a few levels (e.g. the user selects from High, Medium and Low). If medium quality is specified, the system decides video quality so that the video can be played back using all of the amount of remaining battery for remaining time of the video. If high or low quality is specified, the quality is decided by increasing or decreasing the standard playback power calculated for medium quality by predefined rate (e.g. 20%). When the user changes quality specification, or predefined time passes since last change, the system updates the standard playback power.

3.3.2 Grouping quality requests

Video quality in which each user node receives can be calculated by the above method , but transcoding video too many different quality is not desirable in terms of processing power. Thus, in the proposed method, similar video qualities are grouped into a single video quality. This can be achieved by following steps. (1) Permissible difference range r of quality is specified to requested quality $q_u.s, q_u.f, q_u.b$ of each mobile node u, where r is calculated from restrictions to user's satisfaction rate. For example, if satisfaction rate of a user is 0.95, permissible difference range r is $1 - 0.95 = 0.05$. (2) Let S be a set of all quality requests. (3) For each mobile node u, a set of quality requests S_u is calculated so that a quality request $q_{u'} = (q_{u'}.s, q_{u'}.f, q_{u'}.b) \in S$ is an element of S_u if and only if $(1-r)\cdot q_u.s \leq q_{u'}.s \leq q_u.s \wedge (1-r)\cdot q_u.f \leq q_{u'}.f \leq q_u.f \wedge (1-r)\cdot q_u.b \leq q_{u'}.b \leq q_u.b$. (4) Find the set with maximum number of elements, and exclude elements from S. (5) The steps (3) and (4) are repeated until S becomes empty.

3.3.3 Video delivery protocol

The protocol consists of the part before starting video transfer, the part to reconstruct service paths, the part used when a node joins or leaving the group, and the part used in handoff of a node between APs. First of all, we describe the protocol used before starting video delivery.

Starting video delivery

1. Let t be the starting time of video delivery. Before t, each mobile node u whose user wants to watch the video sends quality request q_u calculated by the method described in Sect. 3.3.1 to the connected proxy p.

2. each proxy p sends received requests to the content server s.

3. s does a grouping of all received requests by the method described in Sect. 3.3.2, and it decides the set of qualities $E(p)$ to which p performs transcodings. Let $q_p.s$ and $q_p.f$ be the largest screen size and the largest frame rate in $E(p)$, respectively. p receives a video stream with quality equal to or better than $q_p.s, q_p.f, q_p.b$ from the upstream proxy. Bitrate of q_p can be calculated from screen size and frame rate by the method in [28]. p can now transcode this video stream to ones with any element in $E(p)$.

4. s finds a set of service paths from received video qualities by the algorithm described in Sect. 3.4. s sends a message with the set of service paths to all proxies along the service paths. Each proxy starts all transcoding services and forwarding services after receiving the message.

5. At the time t, server s starts transferring video stream along the service paths. Transcoding services transcode received video to the specified quality, and forwarding services relays video stream to their downstream proxies.

Reconstruction of service paths

Let t_r be the time to reconstruct the service paths. t_r is a boundary between video segments if pre-recorded video is transferred. In the case of live video, service paths are reconstructed periodically. We assume that t_r is informed to all mobile nodes beforehand.

When the time $t_r - \delta$ approaches, each proxy sends received requests to the content server s, where δ is the time required to gather all quality requests, calculate new service paths, distribute them to the all proxies and receive video stream with transcoding delay. s calculates new service paths from received requests by the algorithm described in Sect. 3.4, and sends them to proxies

along the old service paths. All proxies start to transfer video stream along new service paths. Each proxy receives video streams along the old service paths while simultaneously receiving other video streams along the new paths, and stops receiving from the old paths when finishes reconstruction.

If a proxy near the end of a service path moves to near the content server in a new service path, video playback can be interrupted for a while due to transcoding delay. This can be avoided by simultaneously receiving video streams along the old path and the new path for a while. Since the amount of buffer differs in proportion to the number of hops from the content server, this should be adjusted according to new service paths.

Joining and leaving of a node

A joining node u_{new} decides video quality $q_{u_{new}}$ by the method described in Sect. 3.3.1. u_{new} sends *join* message including $q_{u_{new}}$ to the connected proxy p. p chooses a video quality close to $q_{u_{new}}$ from qualities to which p is transcoding, and transfers the video to u_{new}. The video quality is optimized at the next time of service path reconstruction.

If a mobile node u_{leave} wants to stop receiving video, it can leave anytime. If the corresponding proxy has no other mobile nodes receiving video of the quality at which u_{leave} were receiving, its transcoding service is stopped. Accordingly, the quality of video at which p receives from upper proxy can be changed. We will describe how to cope with this situation in Sect. 3.4.

Handoff of mobile node between APs

Each mobile node can move from the range of an AP to the range of another AP. In this case, proxy p compares requested quality $q_{u_{new}}$ of the new node u_{new}, and the quality q_p at which p is receiving from its upstream proxy. If $q_{u_{new}}.s \geq q_p.s \wedge q_{u_{new}}.f \geq q_p.f \wedge q_{u_{new}}.b \geq q_p.b$, it is impossible to instantaneously starting sending video streams at the quality $q_{u_{new}}$, and thus p temporarily sends video of q_p to u_{new}. Video quality will be optimized at the next time of service path reconstruction.

If $q_{u_{new}}.s \leq q_p.s \wedge q_{u_{new}}.f \leq q_p.f \wedge q_{u_{new}}.b \leq q_p.b$, either of the followings are performed.

- if $q_{u_{new}} \in E(p)$, p simply sends an existing stream to u_{new}, where $E(p)$ is the set of qualities to which p performs transcodings. Otherwise, if there is remaining processing power, a new transcoding service is started, and a video stream of $q_{u_{new}}$ is sent to u_{new}. If there is no remaining processing power, the quality nearest to u_{new} is chosen from $E(p)$, and sent to u_{new}.

- The proxy chooses the element closest to $q_{u_{new}}$, and transfers it.

Let D_p be the delay (or latency) of the service path from server to u, where u is receiving video from proxy p. Video can be played back seamlessly if $D_p \leq D_{p'}$, where p' is the proxy for u after handoff. However, if $D_p > D_{p'}$, there can be skip of video playback due to transcoding delay of $D_p - D_{p'}$. This can be avoided by buffering video data at each proxy similarly to the process of service path reconstruction.

As a mobile node u moves, its AP and the corresponding proxy changes. If any mobile nodes are not connected to the new proxy, u does not receive any video from the proxy.

In order to cope with this problem, we slightly extend the algorithm as follows.

Let $NB(p)$ denote the set of proxies whose APs are neighboring to p's corresponding AP. If $R(p) \neq \emptyset$, for each $p' \in NB(p)$ such that $R(p') = \emptyset$, we set $R(p') = \{max(R(p))\}$. For proxies in $NB(p')$, we do not apply this modification recursively. By this extension, whenever u's AP changes, it can receive the required quality video. In this case, video data stream has to be sent faster than playback speed in order to absorb the difference.

3.4 Service Path Construction Algorithms

In this section, we describe algorithms to calculate efficient service paths whose objective function defined in Sect. 3.2 is as small as possible. The inputs of algorithms are topology information of a given overlay network and the quality of video $q_p = (q_p.s, q_p.f, q_p.b)$ which each proxy p must receive from its upstream proxy (see Sect. 3.3.3). Note that q_p is decided as the maximum quality requirement of user nodes connecting to p. These algorithms are executed on the server s, and its output is distributed to proxies in a way similar to that described in

Sect. 3.3.3. The objective function is the weighted sum of the consumed computation power and the consumed network bandwidth. However this minimization has a tradeoff. In order to minimize the total computational power, the number of transcoding services has to be minimized. In this case, however, if many users requesting the same quality video are distributed among different proxies, it may consume a lot of network bandwidth to deliver the video to those users. On the other hand, if we try to minimize the sum of the consumed network bandwidth, many transcoding services may have to be executed to provide various quality videos to user nodes. Finding the optimal solution to this problem is a combinatory optimization problem (i.e., NP-hard). So, we have to design a heuristic algorithm to solve this problem. Consequently, we adopt a policy to extend the existing heuristic algorithm to construct minimal spanning tree (Steiner tree) proposed in [29]. In Sect. 3.4.1, we will describe a basic algorithm which generates a set of service paths from a Steiner tree calculated by the method in [29]. Then, we describe two algorithms which minimize the sum of consumed network bandwidth and the sum of computation power respectively. Finally, we describe a hybrid algorithm based on these algorithms in Sect. 3.4.2.

3.4.1 Calculating service paths from Steiner tree

We call a proxy p a parent proxy of p', if p' is directly receiving video streams from p. p' is called a child proxy of p, if p is a parent proxy of p'. We call a proxy directly receiving streams from the server s a root proxy. We call a proxy which does not have a child proxy a leaf proxy.

The algorithm described in this section calculates a Steiner tree which minimizes the sum of hop counts of overlay links based on the algorithm in [29]. Since all overlay links on the calculated tree have to satisfy constraints (13) and (14) in Sect. 2.2.2, qualities of the received video streams by each proxy are adjusted. This process consists of following four steps.

Step1. Leaf proxy p sends message r_q which includes quality request q_p (i.e.,maximum quality requirement of user nodes connecting to p) to its parent proxy p'.

Step2. When p' receives the messages from all of its child proxies, it compares each received quality $q_p$with its own quality request $q_{p'}$. If $q_p.s \geq q_{p'}.s \vee q_p.f \geq$

$q_{p'}.f \vee q_p.b \geq q_{p'}.b$, it adjusts $q_{p'}$ so that
$q_{p'} = (max(q_{p'}.s, q_p.s), max(q_{p'}.f, q_p.f), max(q_{p'}.b, q_p.b))$.
Next, p' sends message r_q which includes quality request $q_{p'}$ to its parent proxy.

Step3. Step 2 is repeated until the message reaches a root proxy.

Step4. The root proxy sends r_q to the server s.

Computation Power Minimization Algorithm

The case that the total sum of consumed computation power at proxies is minimized (i.e., $\alpha = 1$ in objective function (17)), is that only one transcoder is executed for each quality vector q in a proxy among all proxies. If some mobile nodes have the required quality q and they connect to the proxy p which does not execute any transcoder for q, then, as shown in Fig. 10(a), another proxy p' executing a transcoder for q must forward the video to p so that the mobile nodes can receive the video with q.

Also, if we let transcoders running on each proxy to use the same decoded video and encode it to multiple videos with different quality, the totally consumed computation power at the proxy will be less than they use decoded videos with different quality.

So, this algorithm uses as small number of proxies as possible, to output videos with quality vectors requested by all mobile nodes. Since this problem is combinatory optimization problem, the algorithm uses the following heuristics to simplify the calculation.

1. sort the set of proxies P in decreasing order of their available computation powers. Let $SP = (sp_1, ..., sp_{np})$ denote the sorted list.

2. sort the set of quality requirements from mobile nodes in increasing order of their required computation power (the required computation power for q is given by
   $r_{encode}(q)$). Let $QR = (qr_1, ..., qr_{nu})$ denote the sorted list.

3. for sp_1, assign as many items in QR as possible, satisfying $c\_avail(sp_1) > r_{decode}(q_{orig}) + \sum_{i=1}^{j} r_{encode}(qr_i)$.

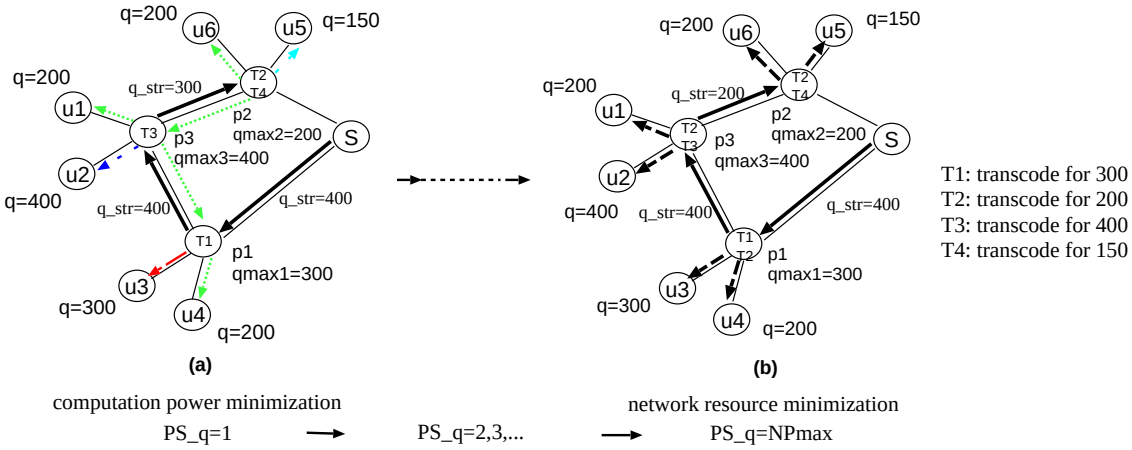

Figure 10. Example of hybrid method

4. similarly, assign as many items as possible to sp_j ($j \geq 2$) from the left items in QR until all items in QR are assigned to proxies.

5. calculate the spanning tree and adjust the maximum quality $p.q$ of each proxy p using the algorithm in Sect. 3.4.1.

Network Resource Minimization Algorithm

The case that the total sum of consumed bandwidths on overlay links is minimized (i.e., $\alpha = 0$ in objective function (17)), is that the same number of transcoders as the number of quality vectors requested by mobile nodes connecting to a proxy p are executed at p. In this case, as shown in Fig. 10(b), each proxy transcodes a video to videos with the quality vectors required by mobile nodes which connect to it. So, redundant video streams are not transmitted between proxies to deliver video with a quality vector q to mobile users in different proxies. As explained in Sect. 3.3.3, each proxy receives the video steam with the highest quality (i.e., maximum picture size and framerate) in the set of quality requirements of mobile nodes. So, it can transcode the video stream to any quality in the set.

3.4.2 Hybrid Method

Let NP_q denote the number of proxies which transcode videos to those with quality q. In the objective function (17), if $\alpha = 1$, then $NP_q = 1$ for all q, and if $\alpha = 0$, then $NP_q = |\{p|q \in E(p) \wedge p \in P\}|$. Let NP_{max} be the maximum value of NP_q in the set of all quality requirements from all mobile nodes.

The problem to minimize the objective function (17) is combinatory optimization problem. So, we use the heuristics that calculate the values of the objective function for all possible values of NP_q between 1 and NP_{max} and select the minimum value among them. Here, we use the same value NP_q for all quality requirements from all mobile nodes.

The algorithm in Sect. 3.4.1 is used to construct the service delivery paths among proxies.

The proposed algorithm is as follows. The Step 2 to Step 4 are repeated for each i from 1 to NP_{max}, and the minimum value of the objective function is selected among them as a solution.

Step1. for each quality vector q, calculate $NP_q = |\{p|q \in E(p) \wedge p \in P\}|$. First, all quality requirements from mobile nodes are divided into multiple groups based on the technique in Sect. 3.3.2. Let $N_{q,x}$ denote the number of mobile nodes requiring quality q at a proxy x. Let PS_q denote the set of proxies which execute transcoders for q. As items of PS_q, i proxies are selected from P in decreasing order of $N_{q,x}$ where $x \in P$.

Step2. Calculate $qmax_x$ which denotes the maximum required quality of mobile nodes at proxy x. $qmax_x$ is calculated by $qmax_x = max(E(x))$.

Step3. Construct a steiner tree among proxies. Based on the algorithm in Sect. 3.4.1, a tree is spanned among proxies with overlay network G and $qmax_x$.

Step4. Construct a steiner tree for each q. If i is larger than 1, i proxies simultaneously transcode and deliver the same quality video to multiple mobile nodes connected to them. So, a steiner tree is constructed to span i proxies for each q. Here, physical hop count is used as cost metrics.

Example

We will give an intuition in the above three algorithms with an example in Fig. 10. In the figure, $qmax_x$ and q_{str} represent the quality which the proxy should

receive from its parent proxy and the quality vector of the stream transmitted through the link, respectively.

Fig. 10 (a) is an example to which the computation power minimization algorithm has been applied. There are six mobile nodes u_1, ..., u_6 and they have either 150, 200, 300, or 400 as their quality requirements (here, we represent quality vectors just as integers for simplicity). In this algorithm, only one transcoding service is executed at a proxy for each quality. So, four transcoding services $T1, T2, T3$ and $T4$ are executed at proxies p_1, p_2, p_3, and p_2, respectively. For example, u_3 requires quality 300 and it is directly connected to p_1, so it can receive the video stream with quality 300. On the other hand, u_4 requests quality 200 and the transcoder for quality 200 is executed at p_2. So, the video stream with quality 200 is transmitted to u_4 via proxies p_3 and p_1. With this algorithm, multiple video streams may be transmitted through each overlay link.

Fig. 10 (b) is an example to which the network resource minimization algorithm has been applied. In this algorithm, each proxy executes transcoding services for mobile nodes which directly connect to the proxy. For example, since u_1 and u_2 directly connect to p_3, p_3 executes two transcoding services for their quality requirements: quality 200 and quality 400. With this algorithm, only one video stream is transmitted through each overlay link.

Our hybrid algorithm minimizes the weighted sum of consumed computation power and consumed network bandwidth represented as the objective function (17) by allowing the both situations simultaneously.

3.5 Performance Evaluation

In order to evaluate effectiveness of our method, we compared three algorithms in the previous section in terms of the achieved cost. The environment of the experiments is as follows: We generated network topologies with 50 proxies using locality model of GT-ITM [30], and used it as the overlay network. We assumed that there are sufficient computational power for proxies and sufficient available bandwidth for links between proxies, in order to compare the costs of outputs from three algorithms. In the experiment, We set the number of user nodes to 2000. We determined physical hop count of each overlay link with a uniform random number between 1 and 10. We set $\tau_d = 0.00057$, $\tau_e = 5 \times \tau_d$. Quality

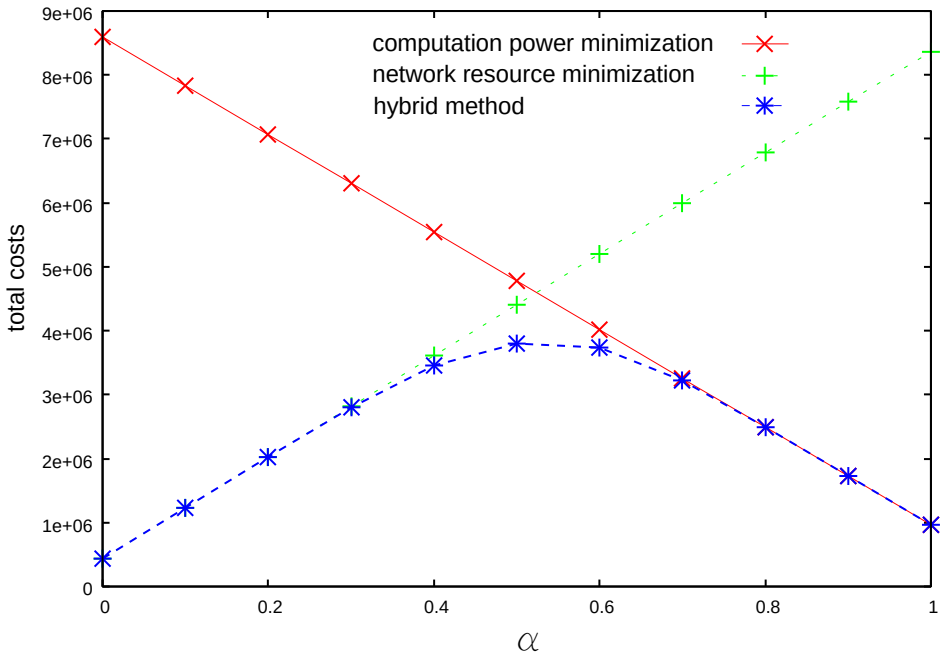

Figure 11. Total costs with different value of α

requirements of the user nodes are generated by uniform random numbers between 80×60 pixel, 5 fps and 640×480 pixel, 30 fps. These are grouped with 20% of permissible difference range. We have measured total costs when α is changed from 0.0 to 1.0. The results are shown in Fig. 11.

Fig. 11 shows that the hybrid algorithm achieve better cost than other two algorithms when α is close to 0.5. The computation power minimization algorithm and the network resource minimization algorithm achieves the minimum total costs when $\alpha = 1.0$ and $\alpha = 0.0$, respectively.

We also measured the performance of the algorithms when the number of user nodes increases. In this experiment, we measured the computation time to generate service delivery paths with 100 to 3000 user nodes. We executed the algorithms on a PC with Athlon 64 3400+ and 1GB RAM. The results are shown in Table 9.

Table 9 shows that the computation power minimization algorithm and the network resource minimization algorithm take almost the same time to complete

Table 9. Time to complete path generation (in seconds)

number of user nodes	100	500	1000	2000	3000
computation power minimization algorithm	0.016	0.12	0.43	1.68	3.91
network resource minimization algorithm	0.023	0.13	0.43	1.67	3.91
hybrid algorithm	0.076	1.34	3.18	9.99	18.7

path generation. The hybrid algorithm takes longer execution time, but the time is practical enough while the number of user nodes is less than 3000.

3.6 Related Work

A lot of literature on service composition has been published so far. [31] treats the performance optimization and security problems in service composition when service components are distributed over ISPs (Internet service providers), and proposes an architecture for efficiently locating and managing service components. [32] proposes a technique to quickly recover from failures on service delivery paths in the wide area network consisting of several ISPs. Our proposed method is different from these studies since it aims at reducing the amount of required resources in the service overlay network, under the given QoS constraints.

Our method is rather related to studies treating network composition problems such as [33, 34, 35, 36]. In [33], in order to provide services with low cost, DAG (directed acyclic graph) is calculated to span multiple service components. Here, available bandwidth and delay of each overlay link between each pair of service instances is given as cost value, and the link with the minimum value is selected. However, this study does not handle the computation power consumed at the proxies. In [34], a service oriented peer-to-peer framework called SpiderNet is proposed, aiming at efficient service sharing among multiple service clients. SpiderNet provides service composition and route selection considering both QoS and node failures. However, it searches each service path independently of other paths. So, multiple paths may conflict in an overlay link when multiple path

searches are activated simultaneously. [35] focuses on defining cost metrics to achieve efficient calculation of service paths by Dijkstra's algorithm considering load balancing based on service replication. [36] proposes a technique for service composition in a service overlay network considering both QoS and resource constraints. Here, Dijkstra's algorithm is used to calculate service paths satisfying constraints. All the above studies treat only service unicast which calculates a single service path for one service user independently, and do not treat service multicast where service components are efficiently shared in multiple service paths to the service users.

[37, 38, 39] propose methods to efficiently deliver multimedia contents to heterogeneous users in various network environments, similarly to our proposed method. However, [37] assumes more constrained environment that the number and the types of service components which can be executed at each proxy are predetermined and do not change. [38] assumes that user requirements are given in advance and do not change. Also, these studies do not handle mobile nodes. [39] proposes an algorithm to calculate efficient service delivery paths by concatenating a multicast tree connecting proxies and local multicast trees consisting of user nodes so that resource consumptions of those trees are minimized. In this method, each local multicast tree is connected to the proxy so that the service delivery path via the proxy has the smaller physical hop count and the larger available bandwidth. However, it does not consider the optimization of resource consumption among multiple service delivery paths on overlay links between proxies nor mobility of user nodes. Our proposed method is different from the above existing studies, since it achieves more flexible service composition where multimedia data can be delivered through the efficient service delivery paths to multiple heterogeneous mobile users whose quality requirements and locations dynamically change.

3.7 Discussion

In Sect. 3.2.2, we defined the objective function in the equation (17) using a parameter α which is given by service provider to make which kinds of resource (computation and network) more expensive. However, it is not clear how to determine the actual value of α based on the current market price of network

and computation devices, management cost of these devices, and so on. At least, we need to normalize network and computation cost so that one unit of network cost corresponds to some units of computation cost. However, when we specify arbitrary function as a normalization function, it may become hard to solve the minimization problem. Thus, in this thesis, we assumed a simple form of the relationship between network and computation cost as defined in the equation (17). As future work, we plan to find an appropriate normalization function and to investigate the weight between network and computation costs in real situations.

As we showed in Sect. 3.5, our hybrid algorithm can save more resource amount compared with simpler algorithms. However, the improvement was less than expected. This is because the algorithm searches only a part of the whole solution space. Thus, by extending the search space, the solution should be improved. In that case, the computation time of the algorithm will be much larger since it is the centralized algorithm. As future work, we plan to develop a decentralized algorithm to make our method more scalable.

3.8 Conclusions

In this chapter, we proposed a service composition based method and algorithms to calculate resource efficient service delivery paths for video multicast to multiple wireless mobile users with different quality requirements. The main contributions of our proposed method are as follows: (1) User's benefit: our method allows heterogeneous mobile users to seamlessly receive and play back video with the required quality which can be dynamically determined based on resource constraints of their mobile nodes such as battery amount, computation power and available network bandwidth, even while they are moving; and (2) Service provider's benefit: service providers can minimize the required resources for the video delivery and limit the resources by giving a dedicated overlay network consisting of a video server, proxies and wireless access points and overlay links among them where only the given bandwidth of each overlay link and the given computation power at proxies are consumed. Through experiments with simulations, we confirmed that our hybrid algorithm can calculate a good approximation of a tradeoff between the consumed network bandwidth and computation power with reasonable

computation time.

4. Conclusion

In this thesis, we have reported our research work on application-level QoS adaptation mechanisms to provide satisfactory multimedia services to users in pervasive environments. We have made the following contributions in this thesis.

First, in Chapter 2, an energy-aware video streaming system for portable computing devices, has been proposed. In the system, playback quality of each video segment is automatically adjusted from the remaining amount of battery, desirable playback duration and the user's preference to each video segment. From some experiments using our system, we confirmed that the playback quality of important video segments can be improved a few times better than flattening the playback quality over the playback duration, and that our system is capable of controlling the playback duration within 6% of error under dynamic control of QoS parameters.

Secondly, in Chapter 3, a new service composition based method for efficient video multicast to multiple mobile terminals, has been proposed. In order to save the resource amount consumed in a CDN, we proposed a heuristic algorithm to calculate service delivery paths which minimize the sum of consumed network and computational resources. Through experiments with simulations, we confirmed that our algorithm can calculate a good approximation of a tradeoff between the consumed network and computation resources with reasonable computation time.

Acknowledgements

First of all, I would like to express my gratitude to Professor Minoru Ito for his support and constant encouragement. This work could not be achieved without his support.

I would like to thank my thesis committee Professor Suguru Yamaguchi for giving me valuable comments and advice.

I am deeply grateful to Associate Professor Keiichi Yasumoto for his kind support and suggestions throughout my whole graduate study. I learned a lot from him about academic thinking, technical writing, presentation skills and how to do research. His advice will be very helpful in my future work.

I would also like to thank Associate Professor Naoki Shibata of Shiga University for his support and valuable technical discussions. I learned a lot from him about technical things such as algorithms and programming techniques.

I would also like to thank Assistant Professor Yoshihiro Murata and Assistant Professor Tomoya Kitani for giving me many valuable comments and suggestions.

I would also like to thank the current and previous members of multimedia group in our laboratory, Sun Tao, Syuuichi Yamaoka and Kazuya Uyama. Working with them has greatly helped me to improve my research. My gratitude also goes to the members of UbiREAL team, Shinya Yamamoto and Hiroshi Nishikawa.

Thanks also go to all current and previous members in our laboratory. Particularly, I would like to thank Koji Nishigaki, Eiichi Takashima and Tatsuhiro Tachibana for their technical and personal support during my graduate life.

I want to give my special thank to Daisuke Ishibe for his support and valuable comments on my research.

Last but not least, I would like to thank my family for their support and encouragement.

References

[1] Dilley, J., Maggs, B., Parikh, J., Prokop, H., Sitaraman, R., and Weihl, B.: "Globally Distributed Content Delivery," *IEEE Internet Computing*, Vol. 6, No. 5, pp. 50–58 (2002).

[2] Vakali, A. and Pallis, G.: "Content Delivery Networks: Status and Trends," *IEEE Internet Computing*, Vol. 7, No. 6, pp. 68-74 (2003).

[3] Pallis, G. and Vakali, A.: "Insight and Perspectives for Content Delivery Networks," *Communications of the ACM*, Vol. 49, No. 1, pp. 101-106 (2006).

[4] Mahesri, A. and Vardhan, V.: "Power Consumption Breakdown on a Modern Laptop," *Proc. of the 4th Int'l. Workshop on Power-Aware Computer Systems (PACS'04)*, LNCS3471, pp. 165–180 (2004).

[5] Agrawal, P., Chen, J.C., Kishore, S., Ramanathan, P., and Sivalingam, K.: "Battery power sensitive video processing in wireless networks," *Proc. of the 9th IEEE Int'l. Symposium on Personal Indoor and Mobile Radio Communications (PIMRC'98)*, pp. 116–120 (1998).

[6] Chang., J. and Tassiulas, L.: "Energy Conserving Routing in Wireless Adhoc Networks," *Proc. of the 19th Annual Joint Conf. of the IEEE Computer and Communications Societies (Infocom 2000)*, pp. 22–31 (2000).

[7] Xu, R., Li, Z., Wang, C., and Ni, P.: "Impact of Data Compression on Energy Consumption of Wireless-Networked Handheld Devices," *Proc. of the 23rd Int'l. Conf. on Distributed Computing Systems (ICDCS'03)*, pp. 302–311 (2003).

[8] Vickers, B.J., Albuquerque, C., and Suda, T.: "Source-Adaptive Multilayered Multicast Algorithms for Real-Time Video Distribution," *IEEE/ACM Trans. on Networking*, Vol. 8, No. 6, pp. 720–733 (2000).

[9] Radha, H.M., van der Schaar, M., and Chen, Y.: "The MPEG-4 Fine-Grained-Scalable video coding method for multimedia streaming over IP," *IEEE Trans. on Multimedia*, Vol. 3, No. 1, pp. 53–68 (2001).

[10] Lin, C.Y., Tseng, B.L., and Smith, J.R.: VideoAnnEx: IBM MPEG-7 Annotation Tool, http://www.alphaworks.ibm.com/tech/videoannex.

[11] Lin, C.Y., Tseng, B.L., Naphade, M., Natsev, A., and Smith, J.R.: "MPEG-7 Video Automatic Labeling System," *Proc. of the 11th ACM Int'l. Conf. on Multimedia*, pp. 98–99 (2003).

[12] The Berkeley MPEG Tools, http://bmrc.berkeley.edu/frame/research/mpeg/.

[13] MJPEG Tools, http://mjpeg.sourceforge.net/.

[14] NOFLUSHD - An idle-disk daemon, http://noflushd.sourceforge.net/.

[15] Lim, J., Kim, M., Kim, J.N., and Kim, K.: "Semantic Transcoding of Video based on Regions of Interest," *Proc. of Visual Communications and Image Processing 2003 (VCIP2003)*, pp. 1232–1243 (2003).

[16] Cavallaro, A., Steiger, O., and Ebrahimi, T.: "Semantic Segmentation and Description for Video Transcoding," *Proc. of the 2003 IEEE Int'l. Conf. on Multimedia and Expo (ICME2003)*, pp. 597–600 (2003).

[17] Pouwelse, J., Langendoen, K., Lagendijk, I., and Sips, H.: "Power-Aware Video Decoding," *Proc. of the 22nd Picture Coding Symposium (PCS'01)*, pp. 303–306 (2001).

[18] Pouwelse, J., Langendoen, K., and Sips, H.: "Dynamic Voltage Scaling on a Low-Power Microprocessor," *Proc. of the 7th annual Int'l. Conf. on Mobile Computing and Networking (MobiCom 2001)*, pp. 251–259 (2001).

[19] Son, D., Yu, C., and Kim, H.: "Dynamic Voltage Scaling on MPEG Decoding," *Proc. of the 8th Int'l. Conf. on Parallel and Distributed Systems (ICPADS'01)* pp. 633–640 (2001).

[20] Choi, K., Cheng, W.C., and Pedram, M.: "Frame-Based Dynamic Voltage and Frequency Scaling for an MPEG Player," *Proc. of the 2002 IEEE/ACM Int'l. Conf. on Computer-Aided Design (CAD'02)*, pp. 732–737 (2002).

[21] Mesarina, M. and Turner, Y.: "Reduced energy decoding of MPEG streams," *Multimedia Systems J.*, Vol. 9, No. 2, pp. 202-213 (2003).

[22] Liu, X., Shenoy, P., and Corner, M.: "Chameleon: Application Level Power Management with Performance Isolation," *Proc. of the 13th annual ACM Int'l. Conf. on Multimedia (MM'05)*, pp. 839–848 (2005).

[23] Shenoy, P. and Radkov, P.: "Proxy-Assisted Power-Friendly Streaming to Mobile Devices," *Proc. of Multimedia Computing and Networking 2003 (MMCN'03)*, pp. 177–191 (2003).

[24] Korhonen, J. and Wang, Y.: "Power-efficient streaming for mobile terminals," *Proc. of the 15th ACM Int'l. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2005)*, pp. 39–44 (2005).

[25] Intel Corporation: Intel XScale Technology Overview, http://www.intel.com/design/intelxscale/.

[26] Intel Corporation: Intel Pentium M Processor Product Information, http://www.intel.com/products/processor/pentiumm/index.htm.

[27] Bernet, Y., Binder, J., Blake, S., Carlson, M., Carpenter, B.E., Keshav, S., Davies, E., Ohlman, B., Verma, D., Wang, Z., and Weiss, W.: "A Framework for Differentiated Services," *IETF working draft <draft-ietf-diffservframework-02.txt>* (1999).

[28] Tamai, M., Sun, T., Yasumoto, K., Shibata, N., and Ito, M.: "Energy-aware Video Streaming with QoS Control for Portable Computing Device," *Proc. of the 14th ACM Int'l. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2004)*, pp. 68–73 (2004).

[29] Kou, L., Markowsky, G., and Berman, L.: "A Fast Algorithm for Steiner Trees," *Acta Informatica*, Vol. 15, No. 2, pp. 141–145 (1981).

[30] Calvert, K.L., Doar, M.B., and Zegura, E.W.: "Modeling Internet Topology," *IEEE Communications Magazine*, Vol. 35, No. 6, pp. 160–163 (1997).

[31] Raman, B., Agarwal, S., Chen, Y., Caesar, M., Cui, W., Johansson, P., Lai, K., Lavian, T., Machiraju, S., Mao, Z.M., Porter, G., Roscoe, T., Seshadri, M., Shih, J., Sklower, K., Subramanian, L., Suzuki, T., Zhuang, S., Joseph, A.D., Katz, R.H., and Stoica, I.: "The SAHARA model for service composition across multiple providers," *Proc. of Int'l. Conf. on Pervasive Computing (Pervasive 2002)*, pp. 1–14 (2002).

[32] Roman, B. and Katz, R.H.: "An Architecture for Highly Available Wide-Area Service Composition," *Computer Communication J.*, Vol. 26, No. 15, pp. 1727–1740 (2003).

[33] Wang, M., Li, B., and Li, Z.: "sFlow: Towards Resource-Efficient and Agile Service Federation in Service Overlay Networks," *Proc. of the 24th IEEE Int'l. Conf. on Distributed Computing Systems (ICDCS 2004)*, pp. 628–635 (2004).

[34] Gu, X., Nahrstedt, K., and Yu, B.: "SpiderNet: An Integrated Peer-to-Peer Service Composition Framework," *Proc. of the 13th IEEE Int'l. Symposium on High-Performance Distributed Computing (HPDC-13)*, pp. 110–119 (2004).

[35] Raman, B. and Katz, R.H.: "Load Balancing and Stability Issues in Algorithms for Service Composition," *Proc. of the 22nd Annual Joint Conf. of the IEEE Computer and Communications Societies (Infocom 2003)*, pp. 1477–1487 (2003).

[36] Gu, X., Nahrstedt, K., Chang, R.N., and Ward, C.: "QoS-Assured Service Composition in Managed Service Overlay Networks," *Proc. of the 23rd IEEE Int'l. Conf. on Distributed Computing Systems (ICDCS 2003)*, pp. 194–201 (2003).

[37] Jin, J. and Nahrstedt, K.: "QoS Service Routing in One-to-One and One-to-Many Scenarios in Next-Generation Service-Oriented Networks," *Proc. of the 23rd IEEE Int'l. Performance Computing and Communications Conf. (IPCCC 2004)*, pp. 503–510 (2004).

[38] Liang, J. and Nahrstedt, K.: "Service Composition for Advanced Multimedia Applications," *Proc. of the 12th Annual Multimedia Computing and Networking (MMCN 2005)*, pp. 228–240 (2005).

[39] Jin, J. and Nahrstedt, K.: "On Exploring Performance Optimizations in Web Service Composition," *Proc. of ACM/IFIP/USENIX Int'l. Middleware Conf. (Middleware 2004)*, pp. 115–134 (2004).

List of Major Publications

Journal Papers

1. Tamai, M., Yasumoto, K., Shibata, N., and Ito, M.: "Battery-aware Video Streaming System to Play Back for Specified Duration on Portable Wireless Devices," *IPSJ Journal* (in Japanese), Vol. 46, No. 4, pp. 1051–1060 (2005).

International Conference

1. Tamai, M., Shibata, N., Yasumoto, K., and Ito, M.: "An Energy-Aware Video Streaming System for Portable Computing Devices," *Proc. of the 7th IEEE Int'l. Conf. on Mobile Data Management (MDM2006)*, CD-ROM proceedings (Demo paper) (2006).

2. Tamai, M., Sun, T., Yasumoto, K., Shibata, N., and Ito, M.: "Energy-aware Video Streaming with QoS Control for Portable Computing Devices," *Proc. of the 14th ACM Int'l. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV2004)*, pp. 68–73 (2004).

3. Tamai, M., Sun, T., Yasumoto, K., Shibata, N., and Ito, M.: "Energy-aware QoS Adaptation for Streaming Video based on MPEG-7," *Proc. of the 2004 IEEE Int'l. Conf. on Multimedia and Expo (ICME2004)*, CD-ROM proceedings (Poster paper) (2004).

4. Tamai, M., Yasumoto, K., Shibata, N., and Ito, M.: "Low Power Video Streaming for PDAs," *Proc. of the 8th Int'l. Workshop on Mobile Multimedia Communications (MoMuC2003)*, pp. 31–37 (2003).

5. Yamaoka, S., Sun, T., Tamai, M., Yasumoto, K., Shibata, N., and Ito, M.: "Resource-Aware Service Composition for Video Multicast to Heterogeneous Mobile Users," *Proc. of the 1st ACM Int'l. Workshop on Multimedia Service Composition (MSC'05) (ACM Multimedia 2005 Workshop)*, pp. 37–46 (2005).

Other Publications

Journal Papers

1. Tamai, M., Shibata, N., Yasumoto, K., and Ito, M.: "Distributed Market Broker Architecture for PC Grid Resource Sharing," *IPSJ Journal* (in Japanese), Vol. 47, No. 2, pp. 455–464 (2006).

2. Sun, T., Tamai, M., Yasumoto, K., Shibata, N., and Ito, M.: "Energy-aware Video Streaming System with QoS Adaptation Based on Inter-segment Importance," *IPSJ Journal* (in Japanese), Vol. 46, No. 2, pp. 546–555 (2005).

International Conference

1. Tamai, M., Shibata, N., Yasumoto, K., and Ito, M.: "Network Simulation Architecture for Smartspace," *Proc. of the 2006 System Support for Ubiquitous Computing Workshop (UbiSys2006)*, CD-ROM proceedings (Poster paper) (2006).

2. Tamai, M., Shibata, N., Yasumoto, K., and Ito, M.: "Distributed Market Broker Architecture for Resource Aggregation in Grid Computing Environments," *Proc. of Cluster Computing and Grid 2005 (CCGrid2005)*, CD-ROM proceedings (2005).

3. Uyama, K., Tamai, M., Murata, Y., Shibata, N., Yasumoto, K., and Ito, M.: "A Delivery Method for Compound Video Playback in Wireless Network," *Proc. of the 2006 Int'l. Workshop on Parallel and Distributed Multimedia Computing (ParDMCom'06)*, LNCS4331, pp. 803–812 (2006).

4. Sun, T., Tamai, M., Yasumoto, K., Shibata, N., Ito, M., and Mori, M.: "MTcast: Robust and Efficient P2P-based Video Delivery for Heterogeneous Users," *Proc. of the 9th Int'l. Conf. on Principles of Distributed Systems (OPODIS2005)*, LNCS3974, pp. 176–190 (2005).

5. Nishikawa, H., Yamamoto, S., Tamai, M., Nishigaki, K., Kitani, T., Shibata, N., Yasumoto, K., and Ito, M.: "UbiREAL: Realistic Smartspace

Simulator for Systematic Testing," *Proc. of the 8th Int'l. Conf. on Ubiquitous Computing (UbiComp2006)*, LNCS4206, pp. 459–476 (2006).

Domestic Conference

1. Tamai, M., Shibata, N., Yasumoto, K., and Ito, M.: "Development of a Network Simulator for Smart Spaces," *Proc. of the 2006 Multimedia, Distributed, Cooperative and Mobile Symposium (DICOMO2006)*, pp. 689–692 (2006).

2. Tamai, M., Shibata, N., Yasumoto, K., and Ito, M.: "A Distributed Architecture for Market-based Resource Allocation in P2P Networks," *Proc. of the 2004 Multimedia, Distributed, Cooperative and Mobile Symposium (DICOMO2004)*, pp. 615–618 (2004).

3. Tamai, M., Shibata, N., Yasumoto, K., and Ito, M.: "Distributed Markets for Market-based Resource Allocation," *Technical Report of IPSJ*, 2003-DPS-115, pp. 39–44 (2003).

4. Uyama, K., Tamai, M., Murata, Y., Shibata, N., Yasumoto, K., and Ito, M.: "A Compound Contents Delivery Method Based on Service Composition in Wireless Environment," *Technical Report of IPSJ*, DPS126/CSEC32, pp. 311–316 (2006).

5. Sun, T., Tamai, M., Yasumoto, K., Shibata, N., Ito, M., and Mori, M.: "Multi-object Video Streaming for Users with Different Preferences," *Proc. of the 12th Workshop on Distributed Processing System (DPSWS-12)*, pp. 95–100 (2004).

6. Sun, T., Tamai, M., Yasumoto, K., Shibata, N., and Ito, M.: "Energy-aware QoS Adaptation for Streaming Video based on Importance of Video Segments," *Proc. of the 2004 Multimedia, Distributed, Cooperative and Mobile Symposium (DICOMO2004)*, pp. 189–192 (2004).

7. Nishikawa, H., Yamamoto, S., Tamai, M., Nishigaki, K., Kitani, T., Shibata, N., Yasumoto, K., and Ito, M.: "UbiREAL: Smartspace Simulator,"

Proc. of the 2006 Multimedia, Distributed, Cooperative and Mobile Symposium (DICOMO2006), pp. 945–948 (2006).

8. Nishikawa, H., Yamamoto, S., Tamai, M., Nishigaki, K., Kitani, T., Shibata, N., Yasumoto, K., and Ito, M.: "A Simulator for Systematic Testing of Ubiquitous Applications in 3D Virtual Space," *Technical Report of IPSJ*, MBL36/UBI10, pp. 239–244 (2006).

9. Yamaoka, S., Sun, T., Tamai, M., Yasumoto, K., Shibata, N., and Ito, M.: "Resource-Aware Video Multicast to Multiple Wireless Mobile Users with Different Quality Requirements," *Technical Report of IPSJ*, 2005-MBL-35, pp. 77–84 (2005).

10. Yamaoka, S., Sun, T., Tamai, M., Shibata, N., Yasumoto, K., and Ito, M.: "An Efficient Video Delivery Method to Various Terminals," *Technical Report of IPSJ*, 2005-DPS-122, pp. 315–320 (2005).