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

Shuichi Yamaoka, Tao Sun, Morihiko Tamai, Keiichi Yasumoto,
Naoki Shibata[†], and Minoru Ito

Graduate School of Information Science,
Nara Institute of Science and Technology
Ikoma, Nara 630-0192, Japan

{shuich-y,song-t,morihi-t,yasumoto,ito}@is.naist.jp

[†] Department of Information Processing and
Management, Shiga University
Hikone, Shiga 522-8522, Japan

shibata@biwako.shiga-u.ac.jp

ABSTRACT

In this paper, we propose a method to deliver video for multiple wireless mobile users with different quality requirements. In the proposed 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 various quality by these services, and delivered to user nodes with the required quality along the service delivery paths. In this paper, 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. The proposed method can treat user mobility where each user node moves to a range of another access point. Also, users of the proposed system can change quality requirements anytime. Through experiments with simulations, we show the usefulness of our method.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Network Architecture and Design—*Network communications,Network topology, Wireless communication*

General Terms

algorithms

Keywords

video streaming, service overlay network, transcoding, mobile nodes

1. INTRODUCTION

Recent innovation and widespread of wireless network technologies have realized various types of portable computing devices (which we call *mobile terminals/nodes*, hereafter), such as laptop PCs, PDAs, and cell phones capable of connecting to the Internet. A lot of networked applications such as Web browsing, e-mail, downloads of high-quality music files, on-line games are available for these mobile terminals. Among them, video streaming is one of the most promising applications. The computation power of micro processors used in mobile terminals is increasing year by year, and it is now sufficient to play back streaming video in real-time. There are wide variety of screen size, computation power, power consumption, battery amount, maximum network bandwidth in these devices. Moreover, data transmission rate and the access point connecting to the Internet changes as they move. From these reasons, in order to broadcast video to multiple mobile terminals simultaneously, the following criteria should be considered and realized: (1) 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 (called *required quality*) to be received; (2) each mobile terminal can change its required quality any time during playback of the video; (3) contents provider should deliver video data of the requested quality; and (4) each mobile terminal can play back video smoothly and continuously even when the access point changes as a consequence of its movement. Letting a server deliver video to multiple mobile terminals by simultaneous unicast streams consumes a lot of computation and network resources. So, (5) it would also be very important to save the resource amount consumed in a content delivery network (CDN) so that the amounts of them are minimized.

There are many research efforts aiming at efficient broadcast of a video to multiple user nodes with different quality requirements. In the most promising technique (e.g., [1]), video data is encoded as a base layer and several extended layers using a hierarchical encoding technique such as MPEG-4 FGS [2] 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. More computation power than decoding a single layered video is also 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. Also, it can only convert bitrate of video. Picture size and frame rate are fixed to the original value.

In this paper, we propose a new service composition based method for efficient delivery of a video to multiple mobile nodes satisfying the above criteria (1) to (5). In the proposed method, we assume the following environments: (i) an overlay network connecting multiple proxies and a video server as shown in Fig. 1 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 [3]; (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.

To achieve the criteria (1) and (3), 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 achieve the criterion (5), 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.

In the proposed method, the above criterion (1) is achieved using our energy-aware video streaming technique proposed in [4]. Here, appropriate quality (i.e., vector of picture size, frame rate, and bitrate) of each segment of the video is automatically determined from (a) 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, and (b) constraints of the mobile terminal consisting of remaining battery amount, available network bandwidth, computation power and screen size. For the criterion (2), we also propose a protocol for periodical reconstruction of new service delivery paths with the latest quality requirements from all mobile nodes. The service delivery paths are reconstructed seamlessly whenever each new video segment is played back. To achieve the above criterion (4), when the current AP (thus the corresponding proxy, too) of a mobile node changes, it immediately starts to receive the video already delivered to the new proxy where the video with the quality closest to (and less than) its required quality is selected. By the next reconstruction of the service delivery paths, the mobile node will be able to receive video with the quality closer to its requirement.

We have implemented the proposed algorithm and measured the consumed resource amount in the overlay network with simulations. As a result, we have confirmed that the proposed method can achieve efficient video delivery to heterogeneous mobile users at low cost, satisfying user's quality requirements.

1.1 Related Work

A lot of literature on service composition has been published so far. [5] 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. [6] 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.

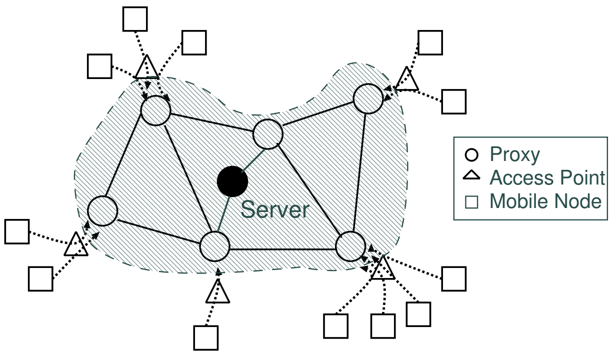

Figure 1: Network environment

Our method is rather related to studies treating network composition problems such as [7, 8, 9, 10]. In [7], 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 [8], 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. [9] focuses on defining cost metrics to achieve efficient calculation of service paths by Dijkstra's algorithm considering load balancing based on service replication. [10] 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.

[11, 12, 13] propose methods to efficiently deliver multimedia contents to heterogeneous users in various network environments, similarly to our proposed method. However, [11] 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. [12] assumes that user requirements are given in advance and do not change. Also, these studies do not handle mobile nodes. [13] 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.

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.

2.1 Target Environments and Assumptions

In our propose method, we assume the existence of a content server, an overlay network, mobile nodes, service components and proxies. We assume the followings:

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. 1). Here, a certain amount of bandwidth is reserved for video delivery on each overlay link using network level QoS techniques such as DiffServ [3]. 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. In this paper, for the sake of simplicity, we treat only the computation power (i.e., CPU usage) required for execution of transcoding services.

2.2 Formal Definition of Problem

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

2.2.1 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. 2 (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 [4]. 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.

$$r_{decode}(q) = \tau_d \times (q.s \times q.f) \tag{1}$$

$$r_{encode}(q') = \tau_e \times (q'.s \times q'.f) \tag{2}$$

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. 2 (b), we show an example of service paths for the overlay network of Fig. 2 (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, $q_i.s \le q_j.s \wedge q_i.f \le q_j.f \wedge q_i.b \le q_j.b$ 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

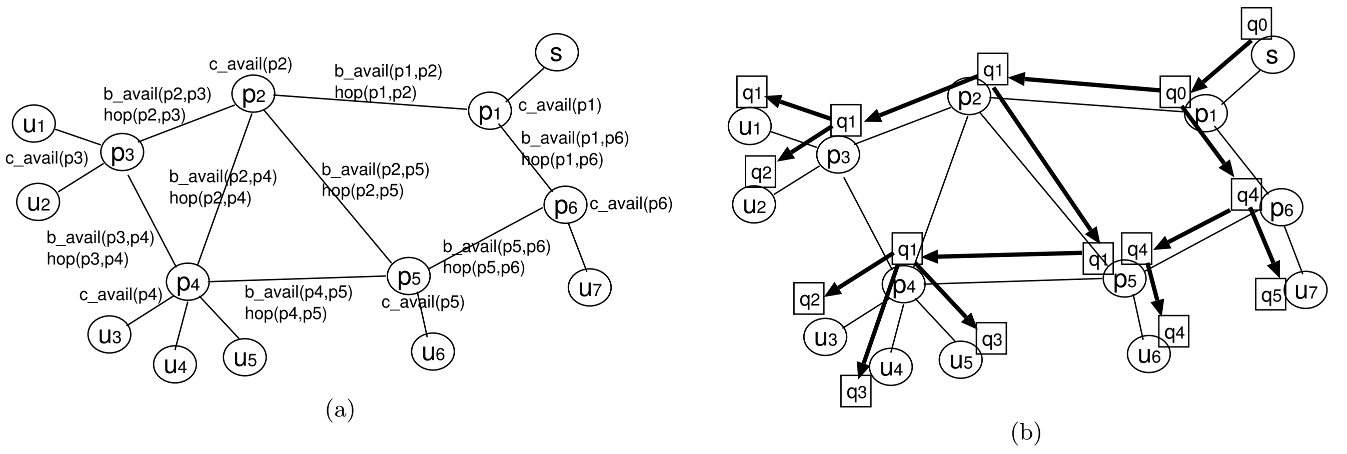

Figure 2: Example of overlay network topology

Fig. 2 (b), $R(p_5) = \{q_1, q_4\}$, $R(u_1) = \{q_1\}$ and $R(s) = \{q_0\}$.

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.

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

$$\begin{aligned} b\_cons(p_i, p_j) = & \sum_{brate \in \{q'.b \mid (p_i,q) \rightarrow (p_j,q')\}} brate \\ & + \sum_{brate \in \{q'.b \mid (p_j,q) \rightarrow (p_i,q')\}} brate \end{aligned}$$

2.2.2 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 $(v', q') \rightarrow (v, q)$ $(v' \in \{s\} \cup P, v \in P \cup U)$, satisfying the following constraints (3)–(6), 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.

$$\begin{aligned} &\text{for each } (v', q') \rightarrow (v, q), \\ &q.s \leq q'.s \wedge q.f \leq q'.f \wedge q.b \leq q'.b \end{aligned} \qquad (3)$$

$$\begin{aligned} &\text{for each } u_i \in U, \\ &\exists(s, p_{j1})\exists(p_{j1}, p_{j2})...\exists(p_{jk}, u_i)\ s.t. \\ &(s, q_{orig}) \rightarrow (p_{j1}, q_1) \ \wedge \ (p_{j1}, q_1) \rightarrow (p_{j2}, q_2) \\ &\wedge \ldots \wedge (p_{jk}, q_{jk}) \rightarrow (u_i, q_{u_i}) \end{aligned} \qquad (4)$$

$$\text{for each } p_i \in P,\ c\_cons(p_i) \leq c\_avail(p_i) \qquad (5)$$

$$\text{for each } (p_i, p_j) \in F,\ b\_cons(p_i, p_j) \leq b\_avail(p_i, p_j) \qquad (6)$$

Constraint (3) 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 (4) 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 (5) and (6) 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.

$$\begin{aligned} &\text{Min } (\alpha \sum_{p \in P} c\_cons(p) \\ &+ (1 - \alpha) \sum_{\{p_i, p_j\} \in F} b\_cons(p_i, p_j) \times hops(p_i, p_j)) \end{aligned} \qquad (7)$$

The first term of the objective function (7) 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. SYSTEM BEHAVIOR

In this section, we describe how the whole proposed system works in detail. In Sect. 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. 3.2, we explain a grouping method of quality requirements to reduce the number of different quality videos for pre-processing. And in Sect. 3.3, we will explain the communication protocol between nodes for video transfer.

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 terminal from time of duration T_u, the amount of remaining battery en_u and constants inherent to the model of mobile terminal (e.g. power consumption of running OS, and so on) in [4]. 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 [14]. 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 [4]. 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.

3.2 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 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.

3.3.1 *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. 3.1 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. 3.2, 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 [4]. 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. 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.

3.3.2 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 transfered. 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. 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.

3.3.3 Joining and leaving of a node

A joining node u_{new} decides video quality $q_{u_{new}}$ by the method described in Sect. 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. 4.

3.3.4 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.

4. SERVICE PATH CONSTRUCTION ALGORITHMS

In this section, we describe algorithms to calculate efficient service paths whose objective function defined in Sect. 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). 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. 3.3.1. 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 [16]. In Sect. 4.1, we will describe a basic algorithm which generates a set of service paths from a Steiner tree calculated by the method in [16]. 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. 4.2.

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 [16]. Since all overlay links on the calculated tree have to satisfy constraints (3) and (4) 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.

4.1.1 *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 (7)), 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. 3(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 $c\_avail(sp_1) > r_{decode}(q_{orig}) + \sum_{i=1}^{j} r_{encode}(qr_i)$.

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. 4.1.

4.1.2 *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 (7)), 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. 3(b), each proxy trancodes 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. 3.3.1, 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.

4.2 Hybrid Method

Let NP_q denote the number of proxies which transcode videos to those with quality q. In the objective function (7), 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 (7) 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. 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. 3.2. Let $N_{q,x}$ denote the number of mobile nodes requiring quality q at a proxy x. Let PS_q denote the

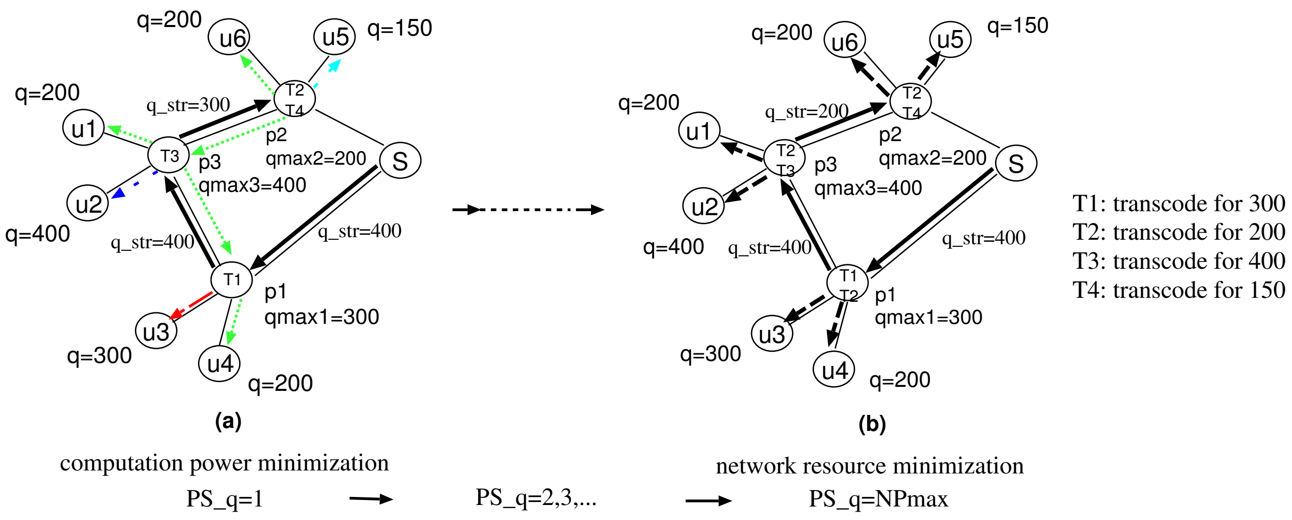

Figure 3: Example of Hybrid Method

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. 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.

4.2.1 Example

We will give an intuition in the above three algorithms with an example in Fig. 3. 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. 3 (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. 3 (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 (7) by allowing the both situations simultaneously.

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, 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 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.4.

Fig. 4 shows that the hybrid algorithm achieve better cost than other two algorithms when α is close to 0.4. 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 1.

Table 1 shows that the computation power minimization algorithm and the network resource minimization algorithm

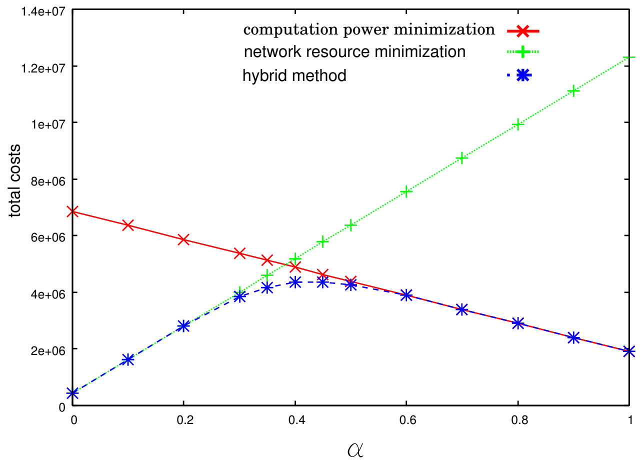

Figure 4: Total costs with different value of α

Table 1: 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

take almost the same time to complete 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.

6. CONCLUSION

In this paper, 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 terminals 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.

As we showed in the previous section, our hybrid algorithm generated slightly better solutions than simpler algorithms. This is because the algorithm searches only a part of the whole solution space (the whole cost computation is done only NP_{max} times). So, 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.

7. REFERENCES

[1] Brett J. Vickers, Celio Albuquerque, and Tatsuya Suda, "Source-Adaptive Multilayered Multicast Algorithms for Real-Time Video Distribution," *IEEE/ACM Trans. on Networking, Vol. 8, No. 6, pp. 720–733*, 2000.

[2] Hayder M. Radha, Mihaela van der Schaar, and Yingwei Chen, "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.

[3] Yoram Bernet, James Binder, Steven Blake, Mark Carlson, Brian E. Carpenter, Srinivasan Keshav, Elwyn Davies, Borje Ohlman, Dinesh Verma, Zheng Wang, and Walter Weiss, "A framework for differentiated services", *IETF working draft <draft-ietf-diffservframework-02.txt>*, 1999.

[4] Morihiko Tamai, Tao Sun, Keiichi Yasumoto, Naoki Shibata and Minoru Ito, "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.

[5] Bhaskaran Raman, Sharad Agarwal, Yan Chen, Matthew Caesar, Weidong Cui, Per Johansson, Kevin Lai, Tal Lavian, Sridhar Machiraju, Z. Morley Mao, George Porter, Timothy Roscoe, Mukund Seshadri, Jimmy Shih, Keith Sklower, Lakshminarayanan Subramanian, Takashi Suzuki, Shelley Zhuang, Anthony D. Joseph, Randy H. Katz, and Ion Stoica, "The SAHARA model for service composition across multiple providers," *Proc. of Int'l. Conf. on Pervasive Computing (Pervasive 2002)*, 2002.

[6] Bhaskaran Roman, and Randy H. Katz, "An Architecture for Highly Available Wide-Area Service Composition," *Computer Communication Journal, 26(15):1727–1740*, 2003.

[7] Mea Wang, Baochun Li, and Zongpeng Li, "sFlow: Towards resource-efficient and agile service federation in service overlay networks," *24th IEEE Int'l. Conf. on Distributed Computing Systems (ICDCS 2004)*, 2004.

[8] Xiaohui Gu, Klara Nahrstedt, and Bin Yu, "SpiderNet: An integrated peer to peer service composition framework," *13th IEEE Int'l. Symposium on High-Performance Distributed Computing (HPDC-13)*, 2004.

[9] Bhaskaran Raman, and Randy H. Katz, "Load balancing and stability issues in algorithms for service composition," *22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003)*, 2003.

[10] Xiaohui Gu, Klara Nahrstedt, Rong N. Chang, and Christopher Ward, "QoS-assured service composition in managed service overlay networks," *23rd IEEE Int'l. Conf. on Distributed Computing Systems*

(ICDCS 2003), 2003.

[11] Jingwen Jin, and Klara Nahrstedt, "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)*, 2004.

[12] Jin Liang, and Klara Nahrstedt, "Service Composition for Advanced Multimedia Applications," *12th Annual Multimedia Computing and Networking (MMCN 2005)*, 2005.

[13] Jingwen Jin, Klara Nahrstedt, "On Exploring Performance Optimizations in Web Service Composition," *Proc. of ACM/IFIP/USENIX Int'l. Middleware Conf. (Middleware 2004)*, 2004.

[14] Ching-Yung Lin, Belle L. Tseng, John R. Smith, "IBM MPEG-7 Annotation Tool," http://www.alphaworks.ibm.com/tech/videoannex

[15] Sumit Roy, Michele Covell, John Ankcorn, and Susie Wee, "A System Architecture for Managing Mobile Streaming Media Services," *Int'l. Workshop on Mobile Distributed Computing (MDC 2003)*, 2003.

[16] L. Kou, George Markowsky, and Leonard Berman. "A fast algorithm for Steiner trees," *Acta Informatica, 15:141–145*, 1981.