FAST ADAPTATION DECISION TAKING FOR CROSS-MODAL MULTIMEDIA CONTENT ADAPTATION

Martin Prangl, Hermann Hellwagner

Klagenfurt University
Department of Information Technology
Universitätsstr. 65–67
9020 Klagenfurt, Austria

Tibor Szkaliczki

Computer and Automation Research Institute
of the Hungarian Academy of Sciences
eLearning Department
Lágymányosi u. 11. H-1111 Budapest

ABSTRACT

In order to enable transparent and convenient use of multimedia content across a wide range of networks and devices, content adaptation is an important issue within multimedia frameworks. The so called Digital Item Adaptation (DIA) standard is one of the core concepts of the MPEG-21 framework that will support the adaptation of multimedia resources according to device capabilities, underlying network characteristics, and user preferences. Most multimedia adaptation engines for providing Universal Multimedia Access (UMA) scale the content with respect to terminal capabilities and resource constraints. This paper focuses on the cross-modal adaptation decision taking process considering the user environment and terminal capabilities as well as resource limitations on the server, network, and client side. This approach represents a step toward increased Universal Multimedia Experience (UME). Based on four different algorithms for solving this optimization process, we present an evaluation of results gained by running their implementations on different test networks.

1. INTRODUCTION

Providing multimedia content over best effort networks like the Internet, through wired and wireless channels, becomes more and more important. Particularly modern terminals like PDAs or mobile phones make it possible to receive the content anytime and anywhere. In order to achieve this goal, often referred to as UMA [1] and targeted at by the MPEG-21 standard, *content adaptation* is necessary to meet the terminal capabilities, network characteristics, and user requirements. But how should the content be adapted to provide the maximum cross-modal utility to the end user? Most adaptive multimedia systems are adapting the multimedia content by simple frame dropping, re-quantization or re-scaling. The challenging question is: “What is the *best adaptation method* under which circumstances?” The answer to this question depends on the content and information assets which a user should collect by consuming the media stream.

This work was in part supported by the Mobile Innovation Center, Hungary

For example, for a given head and shoulder (e.g., news) scene delivered under resource constraints (e.g., bandwidth limitations) it would be better to adapt the video modality in the temporal domain than in the spatial domain and maintain the quality of the audio modality as high as possible. In this manner, the user should get the best multimedia experience (e.g., the news) which the media streams are able to convey under the given constraints. The resulting adaptation decision is further dependent on the available resources (e.g., the computational power) and the individual user preferences.

The development of *utility models* for DIA [2], which form the basis of the cross-modal adaptation decision taking process, is still a challenging research topic [3]. Because DIA is a computationally intensive task in general, the adaptation decision taking process has to be cheap, especially under dynamic resource limitations [4]. Furthermore, the decision may have to be taken every few seconds caused by scene cuts of the delivering media stream.

In this paper, we focus on cross-modal adaptation algorithms, enabling fast decision taking for DIA. After giving a brief introduction into cross-modal utility modelling, we explore the adaptation decision taking problem, considering resource constraints on the server, network, and client side. We will present and discuss four different algorithms for solving this problem. Based on test sequences, we will present the decision results as well as a runtime analysis.

2. CROSS-MODAL UTILITY MODELLING

Cross-modal utility models are needed to estimate the total utility of the modalities delivered in a media stream. A common cross-modal utility approach is to combine the unique modalities by a weighted function [3]. The utility model as used in this paper is detailed in Section 3. The weights are strongly dependent on the type of the content at hand and the individual user preferences. For this reason, the utility model (UM) itself has to be specifically configured on the

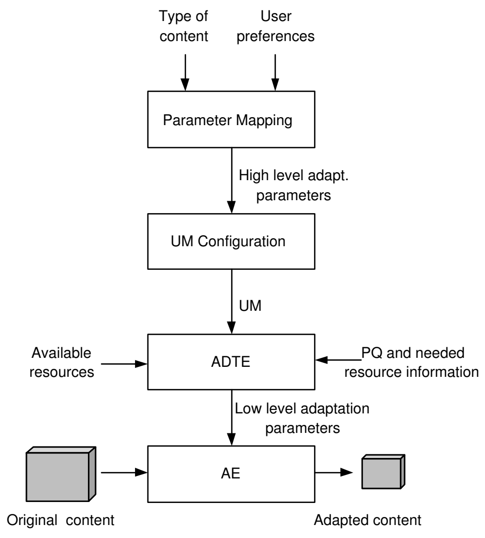

Fig. 1. Concept of cross-modal adaptation decision taking.

media server at runtime by mapping the content type and the user preferences to the utility model parameters, which we call *high level adaptation parameters*. Figure 1 shows an overview of this concept.

In general, the perceptual quality estimation of video or audio scenes is very time consuming. For this reason, the uni-modal utility information as well as the corresponding resources which are needed by the adaptation decision taking engine (ADTE) [2] can be provided as metadata by the MPEG-21 Adaptation QoS (AQoS) descriptor. In our experiments we use PSNR for perceptual video quality estimation [6] and the well known PEAQ metric [7] for audio quality estimation. The information about available resources on the server which has to be estimated locally, as well as the client's available resources which can be provided by MPEG-21 Usage Environment Description (UED) [2] metadata, are also needed by the ADTE for optimal adaptation decision taking. The output of the ADTE are the detailed A/V adaptation parameters (e.g., spatial resolution, frame rate, sample rate, etc.) which we call *low level adaptation parameters*. These parameters are forwarded to the adaptation engine (AE) which performs the adaptation step on the media stream and delivers the specific adapted content to the requesting client.

3. THE PROBLEM MODEL OF DECISION TAKING

A client is requesting movie stream m from the media server. Let the original movie consist of one video and one audio stream, v and a. Both the video and audio streams can be adapted into uniquely defined variations, characterized by a set of video features F_v and a set of audio features F_a. They together form the feature set of a movie, denoted by F_m, which can describe the variations of the movie: $F_m = F_v \cup F_a$. The features can be, e.g., spatial resolution, frame rate, type of codec, number of audio channels, and audio sampling rate. Let features $f_1, f_2 ... f_n$ denote the features ($n = |F_m|$).

Let V_v, V_a denote the sets of deliverable video and audio variations of movie m on the server, respectively. Let V_m denote the set of deliverable variations of movie m. Let M and N denote the number of the different video and audio variations, respectively: $M = |V_v|, N = |V_a|$. The video and audio streams can be combined arbitrarily into a movie, that is, $V_m = V_v \times V_a$. $m|_f$ denotes the value of the feature f of stream m. The variations can be specified as vectors in the feature space: $v = (k_1, k_2 ... k_n)$ where $k_i = v|_{f_i}$. Concatenation of audio and video variations results in the variation of the movie: $v_m = (v_v, v_a)$.

A client request on the movie consists of acceptance sets A_f for each of features $f \in F_m$ which can be acceptance ranges $[f_{min}, f_{max}]$ as well as a special case. Values of the features of the delivered stream m have to fall into the given acceptance sets. $l(f_k)$ is the number of different available and acceptable values of feature f_k.

Furthermore, the utilities of each deliverable video and audio variations are known or can be calculated. Let $U_v(v_v)$ and $U_a(v_a)$ denote the utilities for video variation $v_v \in V_v$ and audio variation $v_a \in V_a$, respectively. As already mentioned, the utility of the multimedia stream resulting from the combination of the video and the audio streams can be calculated as a weighted sum of the utilities of the two modalities: $U(v_m) = w_v \cdot U_v(v_v) + w_a \cdot U_a(v_a)$.

The CPU clock cycles and bit rate needed for each variation are known as well. Let $C_e(v)$, $C_d(v)$, and $B(v)$ denote the encoding and decoding CPU clock cycles and bit rate needs of the variation v, respectively. Trivially, $C_e(v_m) = C_e(v_v) + C_e(v_a)$, $C_d(v_m) = C_d(v_v) + C_d(v_a)$, and $B(v_m) = B(v_v) + B(v_a)$. Furthermore, the CPU usage and the total bit rate of the processed streams are limited on the server. Let L_{C_e}, L_{C_d}, and L_B denote the maximum values of these resources. Let A denote the set of movie variations that satisfy the resource constraints, that is, they fall below the resource limit. These points are called appropriate: $v_m \in A \Leftrightarrow C_e(v_m) \le L_{C_e}$, $C_d(v_m) \le L_{C_d}$, and $B(v_m) \le L_B$.

Our aim is to select a video and an audio variation from the set of the available ones on the server (Eq. 2) such that each of the features of the multimedia stream satisfies the client request (Eq. 6). The CPU requirements of the server and the client have to be considered, the bit rate constraints have to be fulfilled (Eqs. 3, 4, and 5) and the utility of the multimedia stream resulting from their combination has to be maximized (Eq. 1).

Input:

Client request: A_f for $\forall f \in F_m$,

Variations on the server: V_v, V_a,

Limits on bandwidth and CPU usage: L_B, L_{C_e}, L_{C_d}

Output:

Movie variation $v_m = (v_v, v_a)$

Maximize

$$U(v) = (1 - \alpha) \cdot U(v_v) + \alpha \cdot U(v_a) \tag{1}$$

subject to

$$v_v \in V_v, v_a \in V_a \tag{2}$$

$$C_e(v) = C_e(v_v) + C_e(v_a) \leq L_{C_e} \tag{3}$$

$$C_d(v) = C_d(v_v) + C_d(v_a) \leq L_{C_d} \tag{4}$$

$$B(v) = B(v_v) + B(v_a) \leq L_B \tag{5}$$

$$v_v|_f \in A_f, \forall f \in F_v; v_a|_f \in A_f, \forall f \in F_a \tag{6}$$

It can be assumed for most of the features that the resource needs as well as the utility are monotonically increasing while the value of a feature is increasing and the other feature values remain unchanged: $(v_1|_f \geq v_2|_f) \rightarrow U(v_1) \geq U(v_2)$, $C_e(v_1) \geq C_e(v_2)$, $C_d(v_1) \geq C_d(v_2)$, and $B(v_1) \geq B(v_2)$. This is usually true for each video and audio parameter except the video and audio codec type.

4. ALGORITHMS TO SOLVE THE PROBLEM

4.1. All combinations

This approach checks all combinations of the audio and video variations to find the optimum one. This algorithm was implemented in order to validate the results of the other algorithms. The time complexity of the algorithm is $T = O(M \cdot N)$.

4.2. Merging video and audio variations

This algorithm proceeds with video variations according to the increasing order of bandwidth demand while the audio variations are processed in decreasing order. The algorithm is looking for the best utility by generating the combination of the current video variation with the audio variation of the highest utility among those whose resource needs are less than the available resources minus the video resource need.

The algorithm can be efficiently used if the number of different resources is at most two. For this reason, we apply only two resource constraints in the implementation, namely the limits on the bandwidth and encoding CPU. The algorithm manages a subset of audio variations (denoted by T_a) at each step which can participate in an optimum combination with the still unprocessed video variations. The variations are ordered in T_a according to their CPU need.

This method can be used for finding the minimum of a nonlinear global optimization problem which is separable into two groups, that is, the profit (utility) function and the constraints can be written as weighted sums of two variables or variable groups. The algorithm works as follows.

Put video variations v_v and audio variations v_a into lists $L_v()$ and $L_a()$, respectively. Order $L_v()$ and $L_a()$ according to decreasing and increasing bandwidth needs of the variations, respectively. Create empty T_a.

```
$j \leftarrow 1$          //index of audio variations
**for** $i \leftarrow 1..M$ **do**          //index of video variations
  $v_v \leftarrow L_v(i)$
  $B_a \leftarrow B - B(v_v), C_a \leftarrow C - C(v_v)$
  $bInserted \leftarrow true$        // audio variation is inserted
  **while** $j <= N$ **and** $bInserted$ **do**
    $v_a \leftarrow L_a(j)$
    **if** $B_a(v_a) > B_a$ **then** $bInserted \leftarrow false$
    **else**
      Insert $v_a$ into $T_a(j)$, $j \leftarrow j + 1$
      $v'_a \leftarrow$ Predecessor of $v_a$ in $T_a()$.
      **if** $U_a(v'_a) > U_a(v_a)$ **then** Delete $v_a$ from $T_a()$
      **else** $bDeleted \leftarrow true$
      **while** $bDeleted$ **do**
        $v'_a \leftarrow$ Successor of $v_a$ in $T_a()$.
        **if** $U_a(v'_a) < U_a(v_a)$
        **then** Delete $v'_a$ from $T_a()$
          $bDeleted \leftarrow true$     // audio variation is deleted
  Get $v_a$ from $T_a()$ whose CPU need is highest below C.
  **if** $U(v_v, v_a) > maxU$
  **then** $v_m \leftarrow (v_v, v_a)$, $maxU \leftarrow U(v_v, v_a)$
```

For efficiency, the ordered list of candidate audio variations is stored in a so called red-black tree, which is a special balanced tree, where look-up, insertion, and deletion can be done in $O(log(n))$ time (n is the number of nodes in the tree). In this case, the time complexity of the algorithm is $T = O(M \cdot logM + N \cdot logN)$. This can be reduced to $O((M + N)logN)$ if the video variations are ordered in advance according to their bandwidth needs.

4.3. Border scan

This method enumerates the points at the surface of the resource constraints in the joint feature space of all modalities. This algorithm exploits the monotonicity of the utility and resource needs in the feature values.

In the feature space of d dimensions, the resource constraints determine a surface (border) of $d - 1$ dimensions, that separates the movies that comply with each of the resource constraints from the movies that violate any of them. From the monotony of the resource needs it follows that all points below the surface comply with the resource constraints and all points above it do not. From the monotony of the utility it follows that the optimum point is located directly below the border; that is, increasing any of its parameters to the next higher value, if any, results in a variation that needs too much resources. (We call these points as *border points*). As a consequence, it is enough to examine the appropriate points along the border when we search the one with the highest utility. Unfortunately, there is no guarantee on the monotony of the utility along the border, so we have to search the optimum solution at the whole surface of the border.

First, the algorithm looks for a single border point moving in the direction of one selected feature denoted by f_1.

	All combinations	*Merging*	*Border scan*	*Hill climbing*
Runtime (ms)	30.566	36.731 (8.705)	10.332	0.103
Relative utility	100%	100%	100%	99.0%

Table 1. Runtime and utility of the optimization algorithms

Then the algorithm considers the further monotonic features $f_i, (i = 2..n)$ one after the other. For each feature, the algorithm generates the border points restricted to the space of the first i features. Let the set of these points be denoted by B_i. border points in B_i For each border point in B_{i-1}, the value of the current feature f_i is gradually increased while the value of feature f_1 is decreased if necessary in order to create border points in B_i. After considering each feature, we select the border point with the highest utility as optimum. The time complexity of our border scan algorithm is $T = O(M \cdot N/\min(l(f_1), l(f_n)))$.

4.4. Hill climbing

Due to the monotonicity in the resource needs and utility, we could use a heuristic search method, namely steepest-ascent hill climbing [8] as well, and we found it as an efficient approach for real-time application. We start with the worst variation. In each iteration step, we increase the value of the monotonic feature where the utility increase is the highest and the improved variaton still satisfies the resource constraints. The time complexity of the algorithm is: $O(\sum_i l(f_i))$.

5. EXPERIMENTAL RESULTS

We implemented the above algorithms and ran them on several real multimedia stream data. Each input data has the same size: the number of different video and audio variations is 3906 and 12, respectively. Table 1 shows the runtimes of the algorithms and the avarage utilities as percentages of the optimum utility. We observed that each algorithm found nearly optimal solutions. Clearly, heuristic search (hill climbing) is the fastest but it does not always find the exact optimum. Its error was usually less then 2 % but at one test case, it returned with a local optimum whose utility was only 85 % of the global optimum. Merging is the slowest. For this method, we also show the runtime without sorting (time given in brackets), which can be done in advance before client requests arrive. The result indicates that the sorting step takes most of the time of this algorithm; without sorting, this method is the second best. Generating all combinations was not too inefficient because the number of audio variations was small in these experiments.

6. CONCLUSION

Finding video and audio stream variations that maximize the media streams' utility (or the experience) for the user under given resource constraints, represents a complex optimization problem in the multimedia area. We presented and implemented four algorithms to find optimal video and audio variations for cross-modal multimedia content adaptation. We found the simple heuristic hill-climbing optimization method to be the most efficient. However, this algorithm may fail to find the optimum, so further experiments are needed to find (and as a consequence, to avoid) such cases. Other algorithms may be useful in special cases; for instance, the merging method is recommended especially when the utility function is non-monotonic and preparation (sorting) can be done before client requests arrive.

7. REFERENCES

[1] A. Vetro, C. Christopoulos, T. Ebrahimi (eds.), *Special Issue on Universal Multimedia Access, IEEE Signal Processing Magazine*, vol. 20, no. 2, March 2003.

[2] ISO/IEC 21000-7:2004, Information Technology - Multimedia Framework (MPEG-21) - Part 7: Digital Item Adaptation, 2004.

[3] D. S. Hands, "A Basic Multimedia Quality Model", *IEEE Trans. on Multimedia*, vol. 6, no 6., December 2004.

[4] D. Mukherjee, E. Delfosse, J.-G. Kim, and Y. Wang, "Optimal Adaptation Decision-Taking for Terminal and Network Quality-of-Service", *IEEE Trans. on Multimedia*, vol. 7, no. 3, June 2005.

[5] A. Vetro and C. Timmerer, "Digital Item Adaptation: Overview of Standardization and Research Activities", *IEEE Trans. on Multimedia*, vol. 7, no. 3, June 2005.

[6] A.M. Rohaly *et al.*, "Video Quality Experts Group: Current results and future directions", *Proc. of the International Society for Optical Engineering (SPIE) – Visual Communications and Image Processing 2000*, vol. 4067, pp. 742-753, June 2000.

[7] B. Feiten, I. Wolf, E. Oh, J. Seo, and H.-K. Kim, "Audio Adaptation According to Usage Environment and Perceptual Quality Metrics", *IEEE Trans. on Multimedia*, vol. 7, no. 3, June 2005.

[8] E. Rich and K. Knight, *Artifical Intelligence*, MacGraw-Hill, 1991.