Adaptation of Multimedia Presentations for Different Display Sizes in the Presence of Preferences and Temporal Constraints

Yamini Nimmagadda, *Student Member, IEEE*, Karthik Kumar, *Student Member, IEEE*, and Yung-Hsiang Lu, *Senior Member, IEEE*

***Abstract*—Multimedia content adaptation has become important due to many devices with different amounts of resources like display sizes, memories, and computation capabilities. Existing studies perform content adaptation on web pages and other media files that have the same start times and durations. In this paper, we present a content adaptation method for multimedia presentations constituting media files with different start times and durations. We perform adaptation based on preferences and temporal constraints specified by authors and generate an order of importance among media files. Our method can automatically generate layouts by computing the locations, start times, and durations of the media files. We compare three solutions to generate layouts: 1) exhaustive search, 2) dynamic programming, and 3) greedy algorithms. We analyze the presentations by varying screen resolutions, media files, preferences, and temporal constraints. Our analysis shows that screen utilizations are 92%, 85%, and 80% for the three methods, respectively. The time to generate layouts for a presentation with 100 media files is 1200, 17, and 10 s, respectively, for the three methods.**

***Index Terms*—Content adaptation, display size, multimedia presentations, preferences, spatio-temporal layout.**

I. Introduction

ADVANCES in multimedia computing and communication technologies result in the rapid growth of displaying multimedia presentations on a wide range of devices, from mobile phones to workstations. Multimedia presentations may be combinations of media files of different types, resolutions, start times, and durations. To display a multimedia presentation on different devices, an author may have to provide multiple versions of the same presentation according to different devices' resources such as display sizes, computing capabilities, and network bandwidths. Rather than requiring the author to create multiple versions, the content should be adapted automatically. The process of automatic generation of layouts for different devices is called *content adaptation*. We use the terms *media files* and *media objects* interchangeably in this paper. We use the term *authors* for content providers and *users* for viewers. Our method takes inputs specified by the author. No input is required from users because they may not have sufficient knowledge about the content.

In this paper, we present a method for displaying multimedia presentations on screens of different sizes. Instead of creating multiple versions of presentations, the authors specify preferences and temporal constraints on media files. The information is used to automatically generate the presentations suitable for the displays by computing the locations and start times of media files. This automation saves time and human effort in the creation of presentations. The following sections describe the need for automatic generation of presentations based on preferences and constraints.

A. Display Sizes

Mobile devices like cellular phones and personal digital assistants (PDAs) with smaller displays have become popular for viewing multimedia content. At the same time, large displays have gained importance for stationary uses like high-definition televisions, desktops, and informational kiosks in airports. It is difficult for authors to create separate presentations for each display size. Several studies [16], [23], [24], [32], [35] suggest changing the resolutions of media files according to the display size, or converting images and videos to text for smaller displays. The limitations of these studies are: 1) The authors need to provide complete information about the placement of media files on the screens. 2) These studies consider only the content that starts at the same time and ends at the same time.

This paper solves these two problems in the following ways: 1) We select content to fit the display sizes. Our method does not require any inputs regarding the placement on screens. Our method generates this information automatically and places the media files on screens. 2) Since the presentations are also characterized by time, our method ensures that the newly starting media files are placed at suitable times and locations such that the screen areas are utilized efficiently.

B. Preferences

The media files that can be displayed on a larger screen may not fit on smaller screens. On the smaller screens, the content that is more important should be displayed. As the number of media files increases, deciding an order of importance among *all* media files becomes impractical for an author. However, specifying preferences of a media file relative to a few media files is feasible. If a news presentation contains a weather report

Manuscript received July 24, 2009; revised January 30, 2010; accepted May 15, 2010. Date of publication June 07, 2010; date of current version October 15, 2010. This work was supported in part by NSF CNS-0347466 and CCF-0541267. Any opinions, findings, and conclusions or recommendations in the projects are those of the investigators and do not necessarily reflect the views of the sponsors. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Nadia Magnenat-Thalmann.

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47906 USA (e-mail: ynimmaga@purdue.edu; kumar25@purdue.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2010.2052024

Fig. 1. Screen shots of a multimedia presentation. Figure (a) shows four screen shots on a screen of resolution 1600 × 1200, (b) and (c) show four screen shots each of the same presentation on screens of resolution 800 × 600 and 400 × 300. Figures (d)–(f) show the timelines for the three screens. (d) 1600 × 1200, (e) 800 × 600, (f) 400 × 300.

and a commercial, an example of preference is that showing the weather report is preferred to the commercial. Preference elicitation has been studied in decision theory [13], [28] by using utility functions to represent preferences quantitatively. The authors must have the knowledge to specify preferences using these utility functions. In contrast, we develop a preference elicitation tool to specify the preferences in *qualitative* statements. For example, "news is more important than movie; movie is more important than commercial". Our tool dynamically checks consistency. For example, adding the statement "commercial is more important than news" to the previous example results in inconsistency. The preference statements are converted to Trade-off Conditional Preference (TCP) nets [7], [8] for determining media files' importance.

C. Temporal Constraints

Different media files in a presentation may have different durations and start at different times. The times at which the media objects can be displayed is important in some presentations. In order to quantify the temporal order of media objects, we introduce *temporal constraints* in presentations. For example, two commercials can be displayed in a presentation but the first one must end before the second starts. We use a triplet (start time, duration, deadline) to specify a temporal constraint. The duration can be a fixed number or an interval indicating the minimum and the maximum durations. The duration of a video is fixed; the durations of images and text may change. Thus, this paper characterizes multimedia presentations by both temporal and spatial layouts of media files when displayed on different screens.

D. Motivating Example

Fig. 1 shows a multimedia presentation on three different screens of sizes 1600 × 1200, 800 × 600, and 400 × 300. The video clips are creative commons licensed and are downloaded from *http://www.archive.org*. The images are creative commons licensed and downloaded from *flickr*. The presentation is a flash news on a road accident that contains five media files: a video on the accident, an image of a person involved in the accident, text news, and two commercials. The video and the image are available in both high and low resolutions. A full text news is available and a summary of the text news is also available. The first commercial is a technology for vehicle safety. The second commercial is sweaters for viewers in colder places and beach resorts for viewers in warmer places. Table I shows the media files, their resolutions, temporal constraints, and preferences. The preferences indicate that the importance in descending order is video, image, text news, commercial 1, and commercial 2. High-resolution video and image are preferred; full text news is preferred to the summary.

In Fig. 1(a)–(c), each shows four screenshots of the presentation at t = 0, 10, 20, 30 s on the three screens. All the media files can be displayed on the largest screen. However, some files

cannot be displayed on the other two screens. The high-resolution video and the full news do not fit on the second screen, so the low-resolution video and the text summary are displayed. The image and commercial 1 cannot be displayed on the second screen simultaneously. As a result, the image is selected because it is more important. On the smallest screen, only the low-resolution image, the text summary, and the second commercial are displayed. Although the video is more important, it cannot fit and cannot be displayed. Though the first commercial is more important than the second commercial, it cannot be displayed on the smallest screen due to insufficient space. In this example, the viewer of the largest screen is in a colder place and sees the commercial on sweaters. The viewers of other two screens are in warmer places and see the commercial on beach resorts.

Fig. 1(d)–(f) shows the time-line representations of the media files. The rectangular blocks indicate the time at which the media files are displayed. The left (and right) arrows indicate that the files could have started earlier (and ended later). For example, in Fig. 1(d), although C_2 can start at the 10th second, it is displayed from $\mathrm{t} = 20$ s to $\mathrm{t} = 40$ s. The left arrow indicates that C_2 could have started earlier, if it can fit on the screen. The crosses indicate that the other media files cannot be displayed due to lack of space. The durations of the image and the text, and the start times of text are different on the three screens. A media file is placed on the screen when it satisfies three conditions: 1) It can fit on the screen. 2) All the media files that are more important and can fit have been displayed. 3) The file starts on or after the specified start time and can be displayed for at least the minimum duration.

Our previous work [30] presents an adaptation technique to generate presentations when the TCP-nets and constraints are given. This paper extends our previous work in the following ways: 1) We develop a preference elicitation tool to allow authors to specify preferences. This tool checks consistency and converts preferences to TCP-nets. 2) We present three layout generation methods: exhaustive search, dynamic programming, and greedy algorithms. The exhaustive search generates optimal layouts but takes too long. The other two approaches are faster, but they produce suboptimal layouts. 3) We analyze these methods for presentations of different media files, preferences, and temporal constraints.

II. Related Work

Several studies have been conducted on multimedia content adaptation for different devices. Content adaptation can be performed in three ways: 1) content selection, 2) transformation, and 3) layout generation. For smaller screens, important content needs to be selected and transformed to fit on the screens. The layout is then generated based on the information specified by the authors. The following sections provide an overview of the current state of the art for each of these steps.

A. Content Selection

Lee *et al.* [23] classify web pages into blocks including navigation bars, navigation lists, and content. If a block does not fall into any of these categories, it is not displayed on smaller screens. Jan *et al.* [16] use multiple versions of each media file and select a version that is the most suitable for a display. Li *et al.* [24], [27] present *InfoPyramid* as a representation scheme that provides a multimodal and multiresolution hierarchy of multimedia files stored in a server. The layout is entirely characterized by the locations of the media files in these studies. Brafman *et al.* [7], [8] select contents based on authors' preferences and use Trade off Conditional Preference (TCP) nets developed by Boutilier *et al.* [6] for graphical representations of preferences. In a TCP-net, the media files are selected by traversing a preference tree that contains the best solution on the left-most path and the worst solution on the right-most path. However, Brafman *et al.* do not consider automatic layout generation; their method cannot be applied to multimedia files with different start times and durations. This paper presents automatic layout generation and extends TCP-nets to multimedia presentations with temporal constraints.

B. Content Transformation

Transcoding, transmoding, and extraction of hierarchy are the three common transformation techniques. They modify the content to fit on the screen. Transcoding converts images and videos from one resolution to another. Cardellini *et al.* [11] propose a proxy system for transcoding web content. Tang *et al.* [34], Shen *et al.* [31], and Hsiao *et al.* [15] propose streaming media caching algorithms for accelerating transcoding on proxies. Transmoding converts rich media like images and videos to simpler media like text. Several studies [32], [35] present transmoding for universal multimedia access with resource constraints. Megino *et al.* [26] propose a set of virtual camera tools to convert images to videos and videos to images. Extraction of structures is widely used for adapting web content. A hierarchy is constructed based on structures, with higher levels consisting of encompassing objects and lower levels consisting of constituent objects. Many existing studies [4], [18], [19], [25] use Document Object Model (DOM) for extracting hierarchy in web pages. He *et al.* [14] develop an adaptation engine called Xadaptor, which reformats web pages for a given device using structure-level adaptation. Yang *et al.* [36] propose decomposing web pages to construct object trees and render them based on screen sizes. In this paper, we assume that transformed versions of media files are available on the server.

C. Layout Generation

A layout is characterized by both spatial and temporal arrangements for multimedia files. Existing studies consider either spatial or temporal arrangements separately. Borning *et al.* [5] generate layouts for web pages when spatial constraints are given and solved by using linear constraint solvers like Cassowary and BAFSS. Ali *et al.* [1] propose a force-based approach for generating layouts of multimedia documents. They assume attractive force between related files and repulsive force between unrelated files. Kong *et al.* [20]–[22] specify the spatial relationships like direction, alignment, and topology among media files. These studies require the specification of relative positions or constraints on locations of media files. In contrast, our method places media files on the screen based on preferences from the authors. Buchanan *et al.* [9], [10] present a scheduling system called *FireFly* with temporal constraints of media

TABLE I
MEDIA FILES, TEMPORAL CONSTRAINTS, AND PREFERENCES OF A MULTIMEDIA PRESENTATION

Group	Media File	Resolution	Temporal Constraint	Preference
Video	high resolution low resolution	1280 × 720 480 × 360	(0,40,40)	High resolution video is preferred to low resolution video. Video is more important than image.
Image	high resolution low resolution	300 × 600 200 × 300	(0,[20,40],40)	High resolution image is preferred to low resolution image. Image is more important than text news
Text news	full text summary	1400 × 400 300 × 340	(0,[20,40],40)	Full text is preferred to summary. News is more important than commercials.
Commercial 1	safety	200 × 280	(0,20,40)	Commercial 1 is more important than commercial 2.
Commercial 2	sweaters beach resorts	300 × 300 350 × 120	(10,20,40)	Sweaters is preferred to beach resort for viewers in colder places; beach resorts is preferred to sweaters in warmer places.

TABLE II
COMPARISON WITH EXISTING STUDIES. OUR METHOD PRODUCES LAYOUTS MEETING BOTH SPATIAL AND TEMPORAL CONSTRAINTS. ALL THE OTHER METHODS CAN HANDLE ONLY SPATIAL CONSTRAINTS

Method	Media Format	Automatic Layout Generation	Authors' Inputs	Preference
Hsiao et al. [15]	web pages	no	exact locations	no
Jan et al. [16]	web pages	no	exact locations	no
Lee et al. [23]	web pages	no	exact locations	no
Borning et al. [5]	all media types with the same start times and durations	yes	constraints on locations of media files	no
Brafman et al. [8]	all media types (image, video, text) with the same start times and durations	yes	constraints on locations of media files	yes
Buchana et al. [9]	all media types (image, video, text) with the same start times and durations	yes	constraints on locations of media files	no
Our method	multimedia presentations with all media types (image, video, text)	yes	preferences and temporal constraints; locations are generated automatically	yes

files. Jourdan *et al.* [17] present authoring guidelines for specifying temporal constraints based on Allen's temporal interval algebra [2]. Chang [12] propose a temporal algebra system specific to Synchronous Multimedia Integration Language (SMIL). These studies consider the scheduling of media files; however, they do not consider the placement of media files on the screen. In contrast, we consider both the spatial and temporal arrangements of media objects.

D. Our Contributions

The existing studies are limited by their consideration of media files that have the same start times, durations, and deadlines. They also require inputs from authors regarding the placement of the media files. This becomes harder as the number of media files increases. Table II shows the comparison of our method with the existing content adaptation studies. Our method differs from the existing studies in the following ways: 1) We present the adaptation of multimedia presentations with media files of different resolutions, start times, durations, and deadlines. 2) We use preferences to specify the desirable media files and generate an order of importance among the media files using TCP-nets created from the preferences. 3) We present a complete system that performs automatic spatio-temporal layout generation of multimedia presentations in the presence of temporal constraints and preferences.

TABLE III
TABLE OF SYMBOLS

Symbol	Meaning	Section
$D(M)$	domain of the group M	III-A
$\triangleright$	relatively more important than	III-A1
$\longrightarrow$	depends on	III-A1
$\square$	conditionally more important than	III-A1
$\succ$	succeeded in order of selection by	III-A3
$\sim$	equal in order of selection	III-A3
$>$	preferred to	III-A3
r_i	importance of the group containing media file i	III-A3
v_i	preference value of media file i	III-A3
ρ_i	importance of media file i	III-A3
Q	quality of layout	III-B
W	width of the screen (pixels)	III-B
L	height of the screen (pixels)	III-B
w_i	width of media file i (pixels)	III-B
l_i	height of media file i (pixels)	III-B
N	number of media groups	III-B
T	duration of the presentation (seconds)	III-B
T_{s_i}	start time of media file i (seconds)	III-B
T_{d_i}	duration of media file i (seconds)	III-B
T_{e_i}	deadline of media file i (seconds)	III-B
f	clusteredness of TCP-nets	IV-A
d_i	out-degree of a group i in TCP-net	IV-A
g	weighted sum of deadlines with the corresponding number of media files	IV-A
n_t	number of media files with deadline t	IV-A

III. PREFERENCE-BASED CONTENT ADAPTATION

Content adaptation has two steps: 1) selecting content based on preferences and 2) generating layout. The first step involves specifying preferences, representing preferences, and finding an order of importance among the media files, as described in

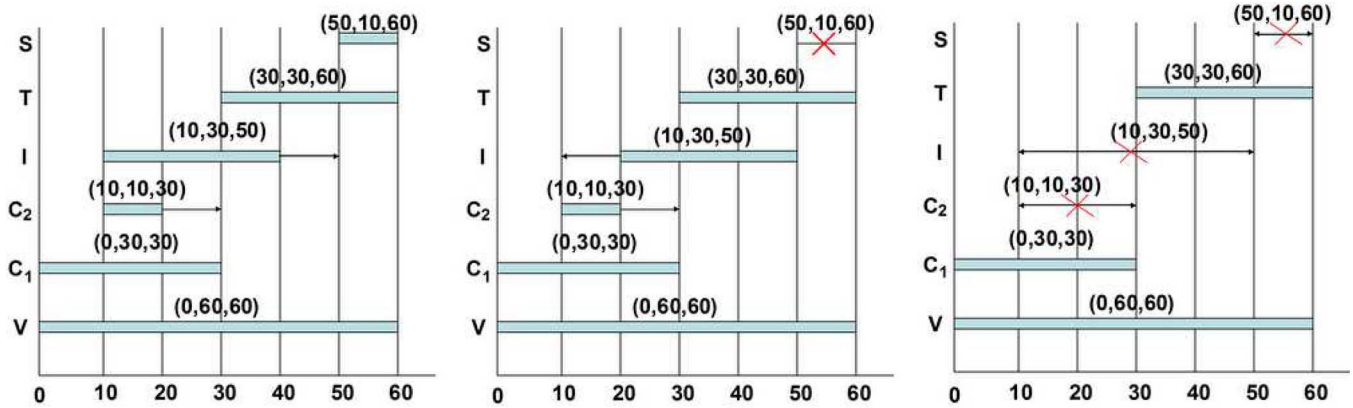

Fig. 2. Timelines of the media files for the three screens. The preferences are specified in Table IV. (a) 1600 × 1200, (b) 800 × 600, (c) 400 × 300.

Section III-A. For the second step, we model layout generation as a constrained optimization problem. Section III-B presents three algorithms for layout generation: exhaustive search, dynamic programming, and greedy algorithm. Table III shows the symbols used in this paper.

A. *Preference Specification, Representation, and Ordering Importance*

A multimedia presentation can be represented as a set $M = \{M_1, M_2, \cdots, M_N\}$. Each M_i is a group and contains one or more media files. At most, one file is selected from each group for display. Preferences may be specified among the files in each group or among the groups. The files in M_i are represented by a set $D(M_i)$, called the *domain* of the group. Table I shows five groups: video, image, text news, commercial 1, and commercial 2. The high-resolution version or the low-resolution version, not both, may be selected for displaying the video. An example of preference within a group is "high-resolution video is preferred to low-resolution video" and an example of a preference among two groups is "video is more important than image". The following paragraphs describe the types of preferences, their representations, and ordering importance among the media files.

1) Specifying Preferences: Preferences convey three types of relationships: 1) relative importance, 2) dependence, and 3) conditional importance.

1) Relative Importance: Consider two domains: films F and commercials C. If a film and a commercial cannot be displayed on a small screen simultaneously, the author may specify the relative importance, "film is more important than commercial", expressed as $F \triangleright C$.
2) Dependence: Dependence describes how selecting one media file influences the selection of the others, for example, if a presentation has two domains: dance videos V and lesson videos L. The first contains two videos, $D(V) = \{\text{salsa, tango}\}$. The second domain has two lessons, $D(L) = \{\text{salsa lesson, tango lesson}\}$. If salsa is selected for the dance video $(V = \text{salsa})$, the salsa lesson is preferred, expressed as $(V = \text{salsa}) \longrightarrow (L = \text{salsa lesson}) > (L = \text{tango lesson})$. Similarly, $(V = \text{tango}) \longrightarrow (L = \text{tango lesson}) > (L = \text{salsa lesson})$. In this example, L depends on V, expressed as $V \longrightarrow L$.
3) Conditional Importance: Conditional importance describes the relative importance between two media files given a condition. For example, the selection of a commercial may depend on the viewer's location l. A commercial on sweaters C_1 is more important than a commercial on beach resorts C_2 for viewers in colder places. The relationship between the two commercials depends on l, expressed as $C_1 \square_l C_2$.

The example in Section I has only relative importance. Table IV shows an example of a presentation on Spanish culture with all the three preference types: relative importance, dependence, and conditional importance. This example has six domains. 1) Video: high-quality on Spanish bullfight V_{B_h}, high quality on Spanish dance V_{D_h}, low-quality on Spanish bullfight V_{B_l}, and low-quality on Spanish dance V_{D_l}. 2) Images: bullfight I_B and dance I_D. 3) Text: full text on bullfight T_{B_f}, summary on bullfight T_{B_s}, full text on dance T_{D_f}, and summary on dance T_{D_s}. 4) Commercial C_1 of Coke: high-resolution C_{1_h} and low-resolution C_{1_l}. 5) Commercial C_2 depends on location L: roller skates C_{2_r} for viewers in warmer locations and snow boots C_{2_s} for viewers in colder locations. 6) Slide show of titles: full titles S_f and summary of titles S_s. Fig. 2(a)–(c) shows the time-line representations of this presentation on screens of sizes 1600×1200, 800×600, and 400×300, respectively.

2) Representing Preferences: The three types of preferences are used for generating TCP-nets and for computing an order of importance. In TCP-nets, each node corresponds to a group of media files. Suppose A, B, and C are three nodes in a TCP-net. There are three types of edges:

1) Relative Importance, $A \triangleright B$: a directed edge from A to B with a triangle.
2) Dependence, $A \longrightarrow B$: a directed edge from A to B without a triangle.
3) Conditional Importance, $A \square_C B$: an undirected edge between A and B, with a weight C.

Each node has a conditional preference table (CPT) for describing the preferences over the media files in the group. In Table IV, the high-quality videos are preferred to the low-quality videos and the video on bullfight is preferred to the video on

TABLE IV
MEDIA FILES, TEMPORAL CONSTRAINTS, AND PREFERENCES OF A MULTIMEDIA PRESENTATION

Group	Media File	Resolution	Temporal Constraint	Preference
Video	V_{B_h} V_{B_l} V_{D_h} V_{D_l}	1280 × 720 480 × 360 800 × 600 128 × 128	(0,60,60)	$V_B > V_D$ $V_{B_h} > V_{B_l}$ $V_{D_h} > V_{D_l}$
Image	I_B I_D	320 × 480 300 × 400	(10,30,50)	$V_B \longrightarrow I_B > I_D$ $V_D \longrightarrow I_D > I_B$
Text	T_{B_f} T_{B_s} T_{D_f} T_{D_s}	1200 × 400 600 × 200 1200 × 420 600 × 200	(20,30,60)	$V_B \wedge I_B \longrightarrow T_B > T_D$ $V_D \wedge I_D \longrightarrow T_D > T_B$ $T_{B_f} > T_{B_s}$ $T_{D_f} > T_{D_s}$
Commercial 1	C_{1_h} C_{1_l}	600 × 200 300 × 100	(0,30,30)	$C_1 \triangleright S$ $C_{1_h} > C_{1_l}$
Commercial 2	C_{2_r} C_{2_s}	320 × 480 300 × 400	(10,10,30)	$T \triangleright C_2$ $C_{2_r} \square_L C_{2_s}$ $L : warmer \longrightarrow C_{2_r} \triangleright C_{2_s}$ $L : colder \longrightarrow C_{2_s} \triangleright C_{2_r}$
Slides	S_f S_s	300 × 700 600 × 100	(50,10,60)	$S \triangleright T$ $S_f > S_s$

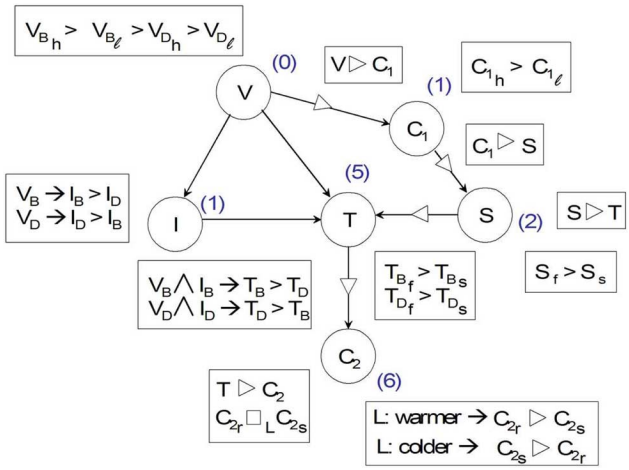

Fig. 3. TCP-net for the multimedia presentation on Spanish culture. The nodes correspond to the groups of media files and the edges correspond to the preferences on groups. The conditional preference tables show the preferences on media files in each group.

dance. This is denoted by $V_{B_h} > V_{B_l} > V_{D_h} > V_{D_l}$. The TCP-net along with the CPTs, shown in Fig. 3, provides complete information about the preferences. We develop a tool to assist authors in specifying preferences. It dynamically checks the consistency of preferences because TCP-nets should be acyclic. If the preferences have no cycle, the tool converts the specifications to TCP-nets.

3) Computing Importance: We compute the importance among the groups first; then we compute the importance of media files in each group. In a TCP-net, "$\sim$" and "$\succ$" denote the order in which media objects are selected. $A \sim B$ means A and B have equal chance of being selected. $A \succ B$ means that A has to be considered for display before B; however, this does not mean A is definitely selected. $A \succ B$ holds for either of the following conditions: 1) A is more important than B. 2) The selection of B depends on the selection of A. The relation $\succ$ satisfies transitivity: if $A \succ B$ and $B \succ C$, then $A \succ C$. A media object is an *ancestor* of another object if the former is considered before the latter. The numbers in the braces in Fig. 3 indicate the number of ancestors for each node.

A group of objects is considered for display only if all its ancestors have been considered. Let $|A(i)|$ be the number of ancestors of the group i. If i and k are two groups and $|A(i)| < |A(k)|$, then group i is more important. If two groups have the same number of ancestors, the group that occupies the larger area on the screen is more important because of a better screen utilization. The area of a group is defined as the largest area among the media files' areas that are smaller than the screen size. Let A_s be the screen size and D_s be the duration of the presentation. The area available for the presentation is the product $A_s D_s$. For each object in group i, we compute the product of this object's area and its duration; then we select the largest product $A_i D_i$ to represent this group. If an object's duration is variable, the minimum duration is used. Thus, each group's importance depends on the number of ancestors and the area occupied $f_i = (A_i D_i)/(A_s D_s)$. Among groups with the same number of ancestors, importance is determined based on the area occupied by the objects on the screen f_i.

The importance among groups with the largest number of ancestors is decided by the areas occupied by them. Hence, the importance r_i for group i with the largest number of ancestors is given by f_i. The importances r_i's for other groups are obtained by adding f_i's to the maximum among the importances of groups with larger numbers of ancestors. This is done to ensure that the importance is greater for the groups with larger numbers of ancestors:

$$r_i = \max_{\forall k}\{r_k\} + f_i, \quad \text{such that } |A(k)| > |A(i)|. \tag{1}$$

In the example shown in Table IV, the order of importance among the groups based on the number of ancestors is $V \succ (I \sim C_1) \succ S \succ T \succ C_2$. I and C_1 have the same number of ancestors. However, I occupies a larger area; hence, $I \succ C_1$. The total order of importance is $V \succ I \succ C_1 \succ S \succ T \succ C_2$. If several objects have equal number of ancestors and occupy the same area on the screen, then these objects have an equal chance to be selected.

After determining the importance among groups, at most one object is selected from each group. The importance among the files in each group is decided by a tree that contains the best

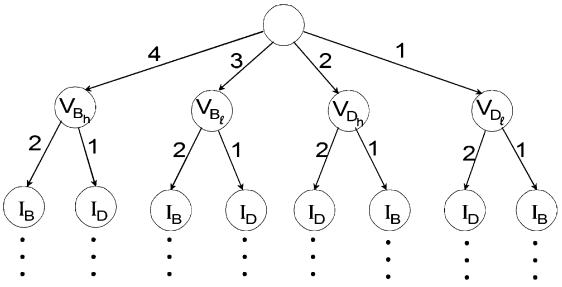

Fig. 4. Tree showing preferences on media files for TCP-net in Fig. 3. The left-most branch indicates the most desired media files.

solution on the left-most branch and the worst solution on the right-most branch. The tree is constructed as follows: 1) The root (depth 0) is empty and does not correspond to any group. 2) The media files in the nth most important group are inserted into the tree at a depth n. 3) At each level, the children of each node are media files arranged in decreasing order of preference from left to right. The preferences among the media files in each group are given by the conditional preference tables. At each level, we assign weights for media files in increasing order from right to left. The right-most node is assigned the weight of one; as we move left, the weight increments by one. The higher the weight, the higher the preference of a media file. Hence, this weight is called *preference value*. The preference value of the ith group is denoted by v_i and is equal to the weight of the media file selected for display.

Fig. 4 shows the tree for the TCP-net in Fig. 3. The most important group is V; hence, the media files in the group V are at depth 1. From the conditional preference table, $V_{B_h} > V_{B_l} > V_{D_h} > V_{D_l}$. They are arranged from left to right in decreasing order of preference. The weights are also assigned in decreasing order of preference: V_{D_l} has a weight 1 and V_{B_h} has a weight 4. The second most important group is I. The media files in I have different preferences for different selections of V. When V_{B_h} or V_{B_l} is selected, $I_B > I_D$; when V_{D_h} or V_{D_l} is selected, $I_D > I_B$. Each file in group V has child nodes from group I. In Fig. 4, the left-most branch $\{V_{B_h}, I_B, \cdots\}$ indicates the best solution. The right-most branch $\{V_{D_l}, I_B, \cdots\}$ indicates the worst solution. The preference value of a media file depends on the selection of other media files. For example, the preference value of I_B is 2, when V_{B_h} or V_{B_l} are selected. I_B's preference is 1, when V_{D_h} or V_{D_l} is selected. In this case, the importance of a media file depends on the selection of other media files. We define the importance of a media file as the summation of the preference value of its group and the preference values of all the groups that are at deeper levels. The importance ρ_i of a media file is therefore given by

$$\rho_i = v_i + \sum_j v_j, \forall j \text{ such that } r_j < r_i. \tag{2}$$

The addition of preference values at deeper levels ensures that the media files at shallower levels of the tree (more important groups) have higher importance values than the others. In Fig. 4, if the selection is $\{V_{B_h}, I_B, \cdots\}$, the importance of media file V_{B_h} is the summation of preference values of V_{B_h}, I_B, and other preference values at deeper levels $(4+2+\cdots)$. The importance value of I_B is the summation of preference values of I_B and preference values of groups present at deeper levels than $I_B(2+\cdots)$. The goal is to maximize the importance of media objects displayed. We use the importance among media files to generate layouts of presentations as described next.

B. Layout Generation

Layout generation is a two-dimensional packing problem with temporal constraints. We present three methods to generate layouts: 1) exhaustive search, 2) dynamic programming, and 3) greedy algorithm. Exhaustive search is pixel-based and accommodates the maximum number of objects by considering all possible arrangements. The other two algorithms are block-based and may produce suboptimal solutions but they are much faster. Once an object is placed in a block, other unused pixels in the block are not considered any more. A layout is *optimal* when the maximum number of important media files is displayed. The following assumptions are used in our layout generation: 1) The temporal constraints are known in advance. 2) The presentation duration is the latest among the deadlines of all media files. We use the term *instant* to describe the minimum unit of time; this paper uses second as the unit. A media file i with start time T_{s_i}, duration T_{d_i}, and deadline T_{e_i} is *ready* at time t if $t \geq T_{s_i}$ and $t + T_{d_i} \leq T_{e_i}$. A media file is *rejected* if it cannot be displayed due to insufficient area on the screen while meeting the temporal constraints.

A block-based technique searches different blocks for placing a media object on the screen. We use Keeping All Maximum Empty Rectangles [3] (KAMER) as the basis because it is a widely-used block-based technique for packing problems. KAMER partitions the unoccupied areas on a screen into maximum empty rectangles (MERs) along the edges of the placed objects. The term "maximum" is used because these rectangles cannot be contained within any other empty rectangle. Each media object is placed at the top left corner in one of the MERs; the choice of the MER depends on the fitting rule used. We adopt a rule to use the screen space more efficiently: an MER is selected for placing an object to reduce the summation of areas of the new MERs formed along the edges of the object that cannot fit any media objects.

Fig. 5 illustrates the placement and removal of three media objects: A, B, and C. In this example, A starts before B, B starts before C, and B ends before C ends. Initially, the entire screen is considered as an MER. In Fig. 5(a), A is placed at the left top corner. The screen is then divided into two new MERs a_1 and a_2, shown in Fig. 5(b) and (c). Since B starts after A, B can be placed in a_1 or a_2. Fig. 5(d) illustrates the placement of B in a_1 and it is divided into two MERs b'_1 and b'_2 along the edges of B. C cannot fit in b'_1 and the wasted space in b'_1 is shown in Fig. 5(e). Fig. 5(f) places B in a_2 and a_2 is divided into two MERs b_1 and b_2. In this case, C can fit; hence, the fitting rule selects MER a_2 for placing B. Fig. 5(g) shows the placement of C in MER a_1 according to the fitting rule. After the placement of C, the MER a_1 reduces to MER c_1 along the edge of C. Fig. 5(h) shows the screen when B ends and C is still displayed on the screen. The MERs b_1 and b_2 are combined back to a_2 along the edge of A.

The quality of a layout is defined as the total importance of the objects selected and displayed. Suppose a presentation has N media groups and duration T. The ith object has importance

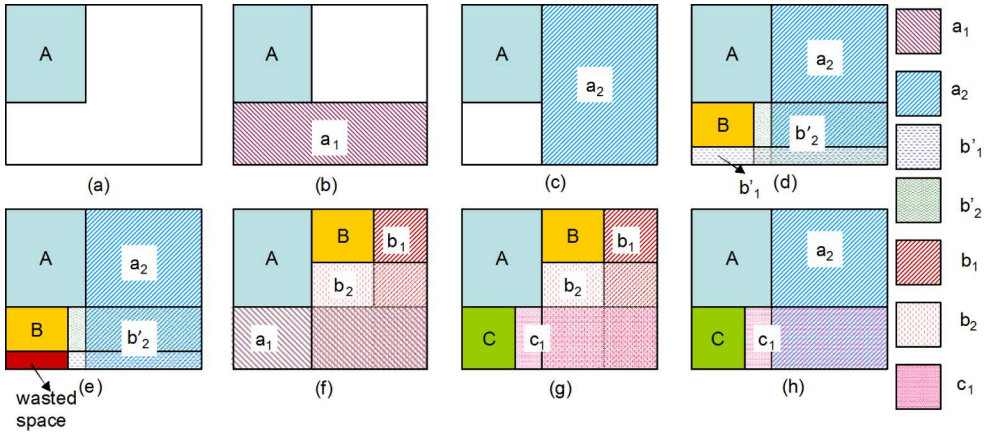

Fig. 5. Example to illustrate KAMER. The rectangles A, B, and C are the media objects. The rectangles with patterns represent MERs. (a) Initially, the screen is an MER; hence, A is placed on the top left corner of the screen. (b) and (c) Screen is partitioned into MERs a_1 and a_2 along the horizontal and the vertical edges of A. (d) Placing B in a_1 partitions a_1 into b'_1 and b'_2. (e) Height of the MER b'_1 is too small for C to fit. The wasted space in b'_1 is shown. (f) Placing B in a_2 partitions a_2 into two MERs b_1 and b_2. (g) Placing C reduces a_1 to c_1 along the edge of C. (h) Removing B combines b_1 and b_2 to obtain MER a_2 again.

ρ_i, width w_i, and height l_i. We want to maximize the objective function Q

$$Q = \sum_{j=0}^{T} \sum_{i=1}^{N} \left(\rho_i c_{ij} \sum_{p \in [0, W - w_i]} x_{ip} \sum_{q \in [0, L - l_i]} y_{iq} \right) \quad (3)$$

where c_{ij} is 1 if the object i is displayed at the jth second; otherwise, c_{ij} is 0. W and L are the width and height of the screen. The variables x_{ip} and y_{iq} are 1 if the upper left corner of object i is located at (p, q) and 0 otherwise. For a presentation with more objects, Q is likely larger; hence, Q is normalized to the maximum attainable value $Q_{\max}$. $Q_{\max}$ is computed by assuming that all the objects can be displayed as per the preferences on a screen of infinite resolution. We need to determine c_{ij}, x_{ip}, and y_{iq}. The objective function Q is nonlinear because of the products of both spatial and temporal variables. The constraints are as follows: 1) Two media objects cannot overlap both temporally and spatially. 2) The media objects should not be displayed before the start time or after the deadline. 3) Any media object can be presented only once and at only one location. 4) A media object must be displayed for at least its minimum duration. We consider three algorithms for layout generation.

1) Exhaustive Search: We consider all possible layouts of objects to achieve the highest quality Q by varying the locations and the start times of objects. Each object can be displayed or rejected depending on the available space on the screen. A presentation with N media groups has 2^N possible selections. Exhaustive search, unfortunately, is time-consuming. For a presentation with 20 objects and a duration of 60 s on a screen of resolution 800×600, there are 60 variables for temporal presence (c_{ij}'s) and 800×600 variables for spatial presence (x_{ip}'s and y_{iq}'s). An upper bound on the total number of variables is determined by assuming that all the pixels are searched for every object. The total number of solutions to be considered is $800 \times 600 \times 60 = 28\,800\,000$ for each object. The possible number of selections of media objects is 2^{20}. The total number of solutions using exhaustive search is $28\,800\,000 \times 2^{20} \approx 3 \times 10^{13}$; this can take many hours. We can reduce the number of possible solutions by searching only unoccupied pixels after some objects are already placed on the screen. Thus, the previous estimation is a loose upper bound. However, exhaustive search is still too time-consuming.

2) Dynamic Programming: This is a bottom-up approach that generates the layouts for presentations with fewer media files and smaller durations first and uses these results to find the layout for a presentation with more media files and longer durations. This method uses a *ready list* and a *rectangle list* to store media files and empty rectangles, respectively. The ready list contains all the media files that are ready at time t. These media files are sorted in descending order of their importance values. The rectangle list contains the maximum empty rectangles that partition the empty space on the screen. These rectangles are determined using the KAMER approach as described in Section III-B. In this approach, a maximum empty rectangle is selected such that the wasted area is reduced.

Let $F(n, t)$ denote the maximum importance for a presentation of $n(\leq N)$ media files and a duration $t(\leq T)$. The placement of n files by time t can be obtained in two ways: 1) one ready media file is added to a presentation of $(n-1)$ files and duration t or 2) the duration of a presentation with n files and duration $t-1$ is increased to t. The values of n and t are dependent and their relationship is captured in the ready list. To find the maximum value of $F(n, t)$, we select the maximum among these two values:

$$F(n,t) = \max \left(F(n, t-1), F(n-1, t) + \max_{i \in \text{ready list}} \{\rho_i\} \right) \quad (4)$$

where ρ_i is the importance of a media file i that belongs to the ready list at time t. The initial values are defined as

$$\begin{aligned} F(0,t) &= 0, \quad \text{if } 0 \leq t \leq T \\ F(n,0) &= F(n-1,0) \\ &\quad + \max_{i \in \text{ready list}} \{\rho_i\}, \quad \text{if } 0 \leq n \leq N. \end{aligned} \quad (5)$$

Algorithm 1 shows the pseudo code of dynamic programming. This algorithm is much faster than exhaustive search; we can further reduce the execution time using a greedy algorithm.

Algorithm 1: Dynamic Programming Layout Generation

```
Input: T, N, ρ, T_s, T_d, T_e
/* Initialization: Presentations with no media files have zero importance */
for t ← 0 to T do
    F[0,t] ← 0;
/* Compute importance of presentations at time t = 0 */
for i ← 1 to N do
    F[i,0] ← F[i-1,0];
    /* The ready list at time t = 0 is stored in Q */
    Q ← ReadyList[0];
    Sort Q in descending order of importances;
    for j ← 1 to length(Q) do
        if Q[j] fits in any of the MERs on screen then
            F[i,0] ← F[i-1,0] + ρ[j];
            Select Q[j] for placement on screen;
            break;
    Place the selected ready object on the screen;
    Partition the screen into MERs;
/* Compute importance of presentations using presentations with fewer media files and smaller durations */
for t ← 1 to T do
    for i ← 1 to N do
        Q ← ReadyList[t];
        Sort Q in descending order of importances;
        for j ← 1 to length(Q) do
            if Q[j] fits in any of the MERs on screen then
                F[i,t] ← F[i-1,t] + ρ[j];
                Select Q[j] for placement on screen;
                break;
        /* Selecting the maximum between presentations with an additional media file and presentations with an additional instant*/
        if F[i,t] > F[i,t-1] then
            Place the selected ready object on the screen;
            Partition the screen into MERs;
        else
            F[i,t] ← F[i,t-1];
```

3) Greedy Algorithm: We propose three versions of a greedy algorithm. The first is a simple solution for selection and placement of media objects in decreasing order of their importance. The other two improve the simple algorithm. All the three algorithms are discussed below.

1) Basic Algorithm: We extend the KAMER algorithm by adding temporal information. At time t, we sort the ready media objects in decreasing order of importance and place them on the screen. This algorithm may reject more important objects that start later due to the lack of space on the screen.
2) Temporal Overlap: We can improve the basic algorithm by considering temporal overlap of media objects and select more important objects even though they start later. Initially, all objects are sorted in descending order of their importance. Before placing an object, we allocate space for the objects that overlap with the object being placed and have higher importance even though they may start later. We define the *overlap factor* $\alpha(i,j)$ as the amount of temporal overlap between two objects i and j. If the two objects overlap $([T_{s_i}, T_{e_i}] \bigcap [T_{s_j}, T_{e_j}] \neq \emptyset)$,, $\alpha_{i,j}$ is the overlapping duration normalized to the sum of the durations:

$$\alpha(i,j) = \frac{\min(T_{e_i}, T_{e_j}) - \max(T_{s_i}, T_{s_j})}{T_{d_i} + T_{d_j}}. \tag{6}$$

If the two durations do not overlap, $\alpha(i,j)$ is zero. Before placing an object on the screen, all the objects that have higher importance and positive temporal overlap with the object are allocated space on the screen. This algorithm considers the importance of all the media files; hence, the quality is higher than the simple greedy algorithm. However, due to allocation of space for temporally overlapping objects before they start, other non-overlapping objects that start later and have higher importance might be rejected. For example, consider a presentation with three media objects A, B, and C with an order of importance $\rho_C > \rho_B > \rho_A$. A starts before B, and B starts before C. Temporal overlap exists between (A, B) and (B, C); there is no overlap between A and C. Before placing A on the screen, space is allocated for B. Even though B and C have positive temporal overlap and $\rho_C > \rho_B$, B is placed before C because the screen is already allocated for B. As a result, C may be rejected due to insufficient space.
3) Temporal Overlap and Back Tracking: Further improvement can be made by tracing back to previously placed objects and releasing the area occupied by less important objects. We control the number of steps of back tracking to prevent tracking all the way to the placement of the very first object. Algorithm 2 shows the pseudo code of the greedy algorithm with temporal overlap and back tracking.

Algorithm 2: Greedy Algorithm with Temporal Overlap and Back Tracking

```
Input: T, N, ρ, T_s, T_d, T_e, Number of allowed back-tracking steps B
/* The set P stores all unplaced media files that are not allocated space on screen. The array 'placed' and 'allocated' store zeroes for unplaced and unallocated objects respectively and ones otherwise.*/
P = NULL;
for i ← 1 to N do
    placed[i] ← 0;
    allocated[i] ← 0;
    Enqueue(P, i);
Sort P in descending order of importances;
Compute temporal overlap matrix α;
for t ← 0 to T do
    Q ← ReadyList[t];
    for i ← 1 to length(Q) do
        if Q[i] fits in any of the MERs on screen then
            if allocated[i] == 0 then
                Place Q[i] on screen;
            else
                Place Q[i] at the allocated space;
            Partition the screen into MERs;
            placed[i] = 1;
            for j ← 1 to N do
                /* Checking for temporally overlapping objects that possess higher importance */ if (i ≠ j and α(i,j) > 0 and ρ[j] > ρ[i]) then
                    Allocate space for j on the screen;
                    allocated[j] = 1;
        else
            /* Tracing back to release space allocated for less important objects */
            d ← 0;
            while d ≤ B do
                Release the space allocated for less important object;
                d++;
                if Q[i] fits in any of the MERs on the screen then
                    Place Q[i] on the screen;
                    Partition the screen into MERs;
                    break;
```

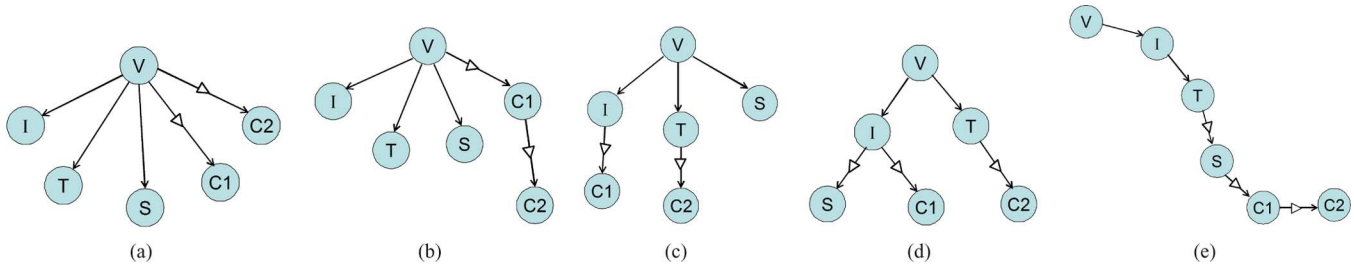

Fig. 6. TCP-nets with different distributions of nodes in descending order of D. (a) clustered, f $= 1$, (b) f $= (16+1)/25 = 0.68$, (c) f $= (9+1+1)/25 = 0.44$, (d) f $= (4+4+1)/25 = 0.36$, (e) diffused, f $= (1+1+1+1+1)/25 = 0.2$.

IV. Evaluation

We use C# to implement and evaluate our methods on an Intel Core 2 Duo processor with 2-GHz speed and 2 GB of RAM. We use Synchronous Multimedia Integration Language (SMIL) to specify the locations, start-times, and durations of objects in presentations. The source code of our algorithms is available at [29]. Existing layout generation methods do not consider temporal constraints of media files in presentations. We adapt some of the widely used two-dimensional packing algorithms [33] to consider temporal constraints and compare our method with these methods. We perform experiments to compare screen utilization, quality of layouts, and layout generation time for these approaches by averaging over 100 different multimedia presentations.

A. Controllable Parameters

An author can specify the following parameters in a presentation: 1) the total number of media files, 2) preferences, and 3) temporal constraints. We evaluate the effects of these parameters independently.

1) Number of media files: To make a fair comparison, we create presentations with more media files by adding new media files, preferences, and temporal constraints to another presentation with fewer files. As we increase the number of media files, we also increase the durations, preferences, and temporal constraints. When analyzing the behavior of the other parameters, we fix the number of media files to be 20.
2) Preferences: A TCP-net can be *clustered* or *diffused*. In a clustered TCP-net, all except one file depend on that one file; this one file has outgoing degree of $N-1$ and the other files have outgoing degree of zero. The importance of each file is decided entirely by the screen area occupied by the media files. A diffused TCP-net has files of smaller outgoing degrees. The clusteredness of a TCP-net can be quantified using $f = 1/(N-1)^2 \sum_{i=1}^{N}(d_i^2)$, where d_i is the number of outgoing edges of object i. The highest value of f is 1 in a clustered TCP-net. As the clusteredness decreases, the value of f decreases. As shown in Fig. 6, a TCP-net with a higher value of f is more clustered, or shallower. A smaller value indicates that the TCP-net is deeper.
3) We use the deadlines to quantify the number of possible choices of selecting media files for display. If most deadlines are near the beginning of the presentation, the choices are limited. In contrast, if most deadlines are near the end of the presentation, more choices are available. Let n_t be the number of objects whose deadlines are at time t. We use $g = 1/NT \sum_{t=0}^{T} tn_t$ to quantify the number of choices. A larger g indicates more choices. The largest value of g is 1 for presentations with all the deadlines at the end of presentation. Our evaluation uses presentations with duration 60 s and media files with a minimum display duration of 5 s.

B. Analysis

1) Screen Utilization: The screen utilization U is the ratio of area occupied by the media files to the total screen area ($U =$ used screen area/total screen area); a higher utilization is preferred. We compute U for screens with different resolutions over 100 presentations with the same duration (60 s), the same number of media files (20 media files), clustered TCP-nets and deadlines at the end of presentations. Fig. 7(a) shows average U for these presentations on screens with different resolutions. U increases first with the screen resolution and then decreases. When the resolution increases, more media files can be displayed on the screen, thus increasing U. Once all media files are displayed, the used area (numerator) saturates and a larger screen increases the denominator, thus decreasing U. The utilizations are very low for resolution 128×128, between 26%–35% for various methods. In contrast, for resolution 1024×768, the utilization increases to approximately 80%. When the screen resolution is fixed, increasing number of media files N may increase U because more options are available. When there are enough files, U remains unchanged. Fig. 7(b) shows the relationship between U and N. U increases from 30% to 90% as N increases from 10 to 80.

Next, we consider how f affects utilizations. For clustered TCP-nets (larger values of f), the media files are selected for display based on their areas occupied on the screen; hence, the screen utilization increases. For diffused TCP-nets, the selection of media files depends on the number of ancestors, not on the occupied screen areas; hence, U is smaller. Fig. 7(c) shows the average values of U versus f. We consider presentations with 20 media files. The value of f ranges from $19/369 = 0.05$ (for diffused TCP-nets) to 1 (for clustered TCP-nets). The screen is utilized only up to 50% for diffused TCP-nets whereas clustered TCP-nets obtain a screen utilization of up to 92%.

U also depends on temporal constraints. When g is large, most files can be placed at any time on the screen, as long as there is enough space; hence, U is higher. We compare U across time for presentations of 20 media files and duration 60 s on a screen of resolution 1024×768. In Fig. 8(a), all the deadlines

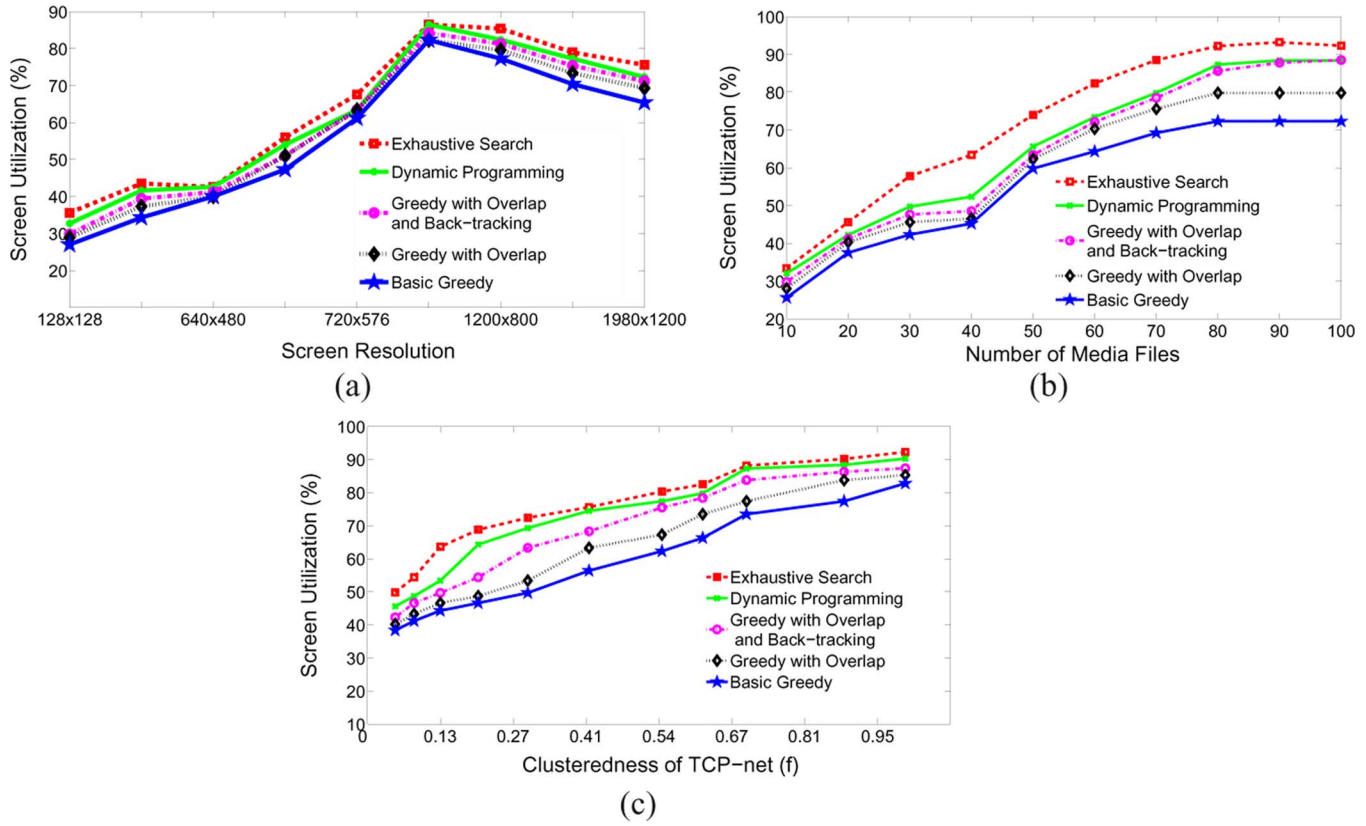

Fig. 7. (a) Screen utilization U for different screen resolutions. Higher U is desired. Exhaustive search has higher U than the other methods. U increases with screen resolutions until all the media files are placed and then decreases. (b) U increases as N increases. (c) U for different f. U increases as f increases because in clustered TCP-nets, media files are selected based their areas and durations. Presentations with 20 media files are used; f ranges between 0.05 (19/369) for diffused TCP-nets and 1 for clustered TCP-nets.

occur at $t = 60$ s. The minimum display duration of any object is 5 s. The utilization for exhaustive search and dynamic programming is almost uniform, whereas U for greedy algorithms is high at the beginning and reduces later because greedy algorithms place media objects as soon as they are ready. Fig. 8(b) shows U when half of the deadlines are at $t = 30$ s and the other half are at $t = 60$ s. In this case, the screen utilization drops substantially after 30 s because half of the media objects can no longer be displayed, even if there is space on the screen. Fig. 8(c) shows U when one fourth of deadlines occur at $t = 15, 30, 45$, and 60 s. The utilization further decreases due to fewer number of choices for placing objects.

As the screen size becomes larger, all the media files can be accommodated on the screen as soon as they are ready to be placed. Hence, the temporal constraints do not play a major role in the selection of locations for large screens. On a screen of infinite size, all the media files are placed at their start-times. By relaxing spatial constraints, temporal constraints do not affect layout generation. Temporal constraints can be relaxed by making the deadlines infinite. The media files can be placed at anytime when the screen can accommodate them. By relaxing spatial or temporal constraints, all the media files can be placed on the screen. However, the screen utilization decreases because the total area increases, whereas the used area saturates after all the media files are placed.

2) Quality of Layouts: We measure the qualities of layouts of presentations by the function Q as described in Section III-B. The value of Q is normalized to the maximum attainable value $Q_{\max}$ (described in Section III-B). Q increases at larger screen resolutions because more media files can be displayed. Q is the highest for exhaustive search, and the least for the greedy algorithm without temporal overlap or back-tracking. Fig. 9(a) shows the qualities of various methods for different screen resolutions. Exhaustive search produces higher quality layouts compared with dynamic programming and greedy approaches. Q increases with the screen resolutions until all the media files are considered and then remains constant. For a screen of resolution 128×128, Q generated by exhaustive search is 41% whereas dynamic programming results in 38% and the three greedy algorithms achieve 23%, 30%, and 31%, respectively. Q reaches one at resolution 1200×800 by exhaustive search; however, dynamic programming and greedy algorithms cannot achieve the maximum quality, even at resolution 1980×1200 due to the inefficient placement of media files.

When the screen resolution is fixed, Q decreases as the total number of media files increase because $Q_{\max}$ increases. For a larger N, the number of media files that cannot be displayed on the screen increases, thereby decreasing Q. Fig. 9(b) shows Q for presentations of different N. Using exhaustive search, Q reaches 85% for $N = 10$ and decreases to 28% for $N = 100$ because more media files are rejected due to insufficient space on screen. Dynamic programming and greedy algorithms follow the same trend and their Q values are slightly lower than those for exhaustive search.

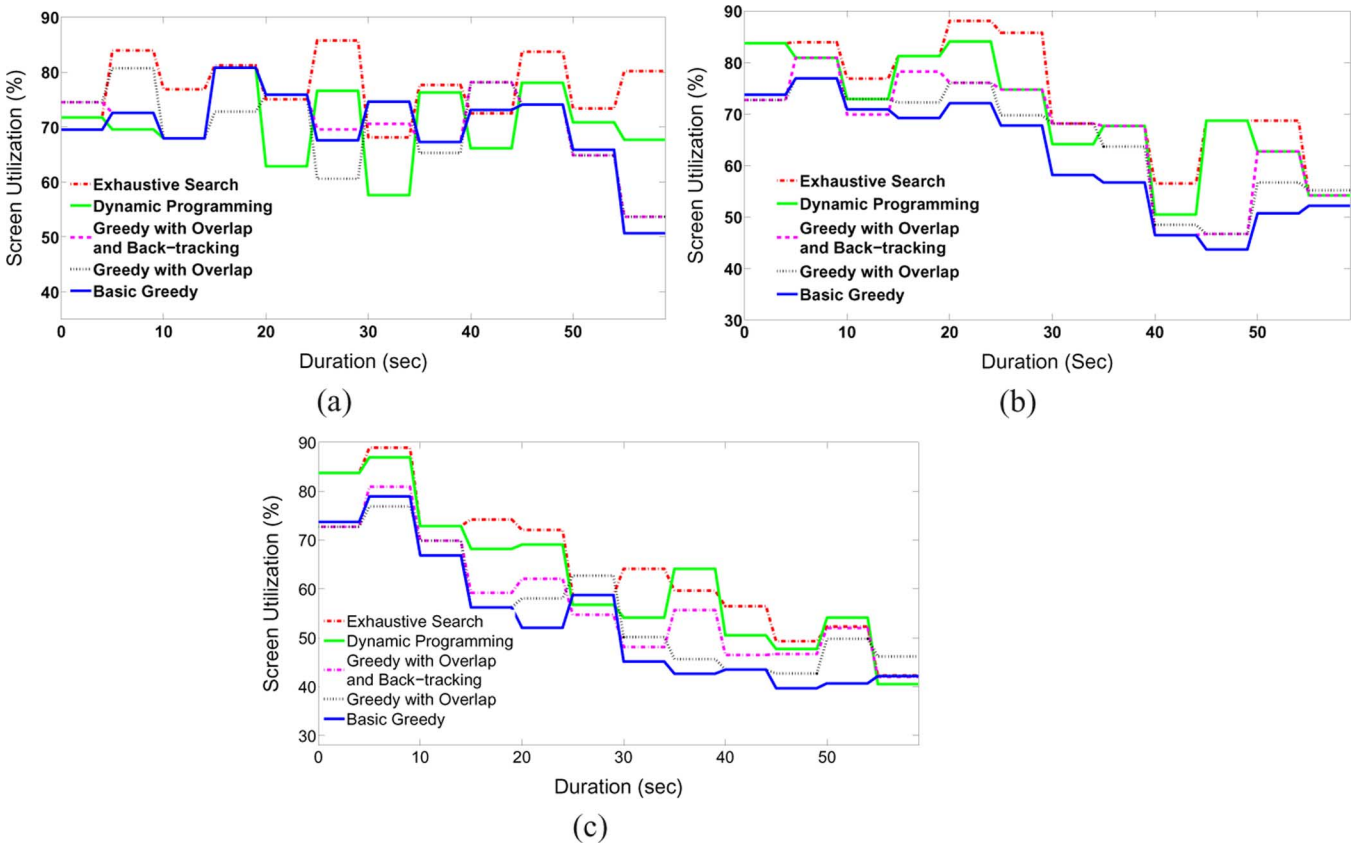

Fig. 8. (a) Screen utilization U for presentations with all the deadlines at $t = T(g = 1)$, (b) half the deadlines at $t = T/2$ and others at $t = T(g = 3/4)$, and (c) one fourth of deadlines are at $t = T/4$, $t = T/2$, $t = 3T/4$ and $t = T$ each $(g = 5/8)$.

Q does not vary much with different f (clusteredness of TCP-nets). However, Q varies with distribution of temporal constraints: Q is higher for larger g. As more deadlines are pushed towards the beginning of presentation, Q decreases as g decreases because fewer objects can be placed on the screen. Fig. 9(c) shows Q for different values of g. The presentations considered in our experiments have 20 media files and durations of 60 s. Exhaustive search produces $Q = 98\%$ when all deadlines occur at the end. Q is 92% for dynamic programming and 80%, 83%, and 87% for the three greedy algorithms. By relaxing spatial or temporal constraints, the quality Q reaches 100%, since all the media files can be placed.

3) Layout Generation Time: A presentation is generated after knowing the screen size. As a result, the viewer has to wait and the execution time is an important factor in comparing algorithms. The execution time of our algorithms depend on the number of media files in the presentation and the screen resolution. The time increases rapidly with a larger N using exhaustive search. The dynamic programming and greedy algorithms are much faster because they use block-based search for locations. Fig. 10(a) shows the layout generation times of various methods for presentations with different numbers of media files on a screen of resolution 1024×768. Exhaustive search takes about 40 s for ten objects and 690 s for 100 objects. The time remains nearly unchanged after a certain number of media files because the screen is occupied and cannot accommodate more objects. Dynamic programming and greedy algorithms take only 110 s and 20 s, respectively, to generate layouts for 100 objects. The time for dynamic programming increases faster with the number of media files compared with the greedy algorithms.

The layout generation time using exhaustive search also increases with the screen resolution because more pixels have to be considered for placing the objects. The time does not increase much for dynamic programming nor greedy algorithms, because the number of blocks formed by partitioning the empty space on screens does not increase significantly even at higher resolutions. Fig. 10(b) shows the layout generation times for different screen resolutions. Exhaustive search takes about 90 s to generate layouts for presentations with 20 media files on a screen of resolution 128×128 and 1200 s for the same presentations on a screen of resolution 1980×1200. Dynamic programming takes 17 s and the three greedy algorithms take 7 s, 8 s, and 10 s for the same presentations on screens of all sizes. The time remains almost the same, even at higher resolutions for dynamic programming and greedy methods, because only the blocks along the edges of placed objects are searched for placement. The clusteredness of TCP-nets and temporal constraints do not affect the execution time significantly. Relaxing spatial or temporal constraints increases the execution time of exhaustive search tremendously because of more choices for placing media files.

4) Comparison With Existing Packing Algorithms: Existing packing algorithms do not consider temporal constraints; their main goal is to minimize the total unoccupied screen area. In

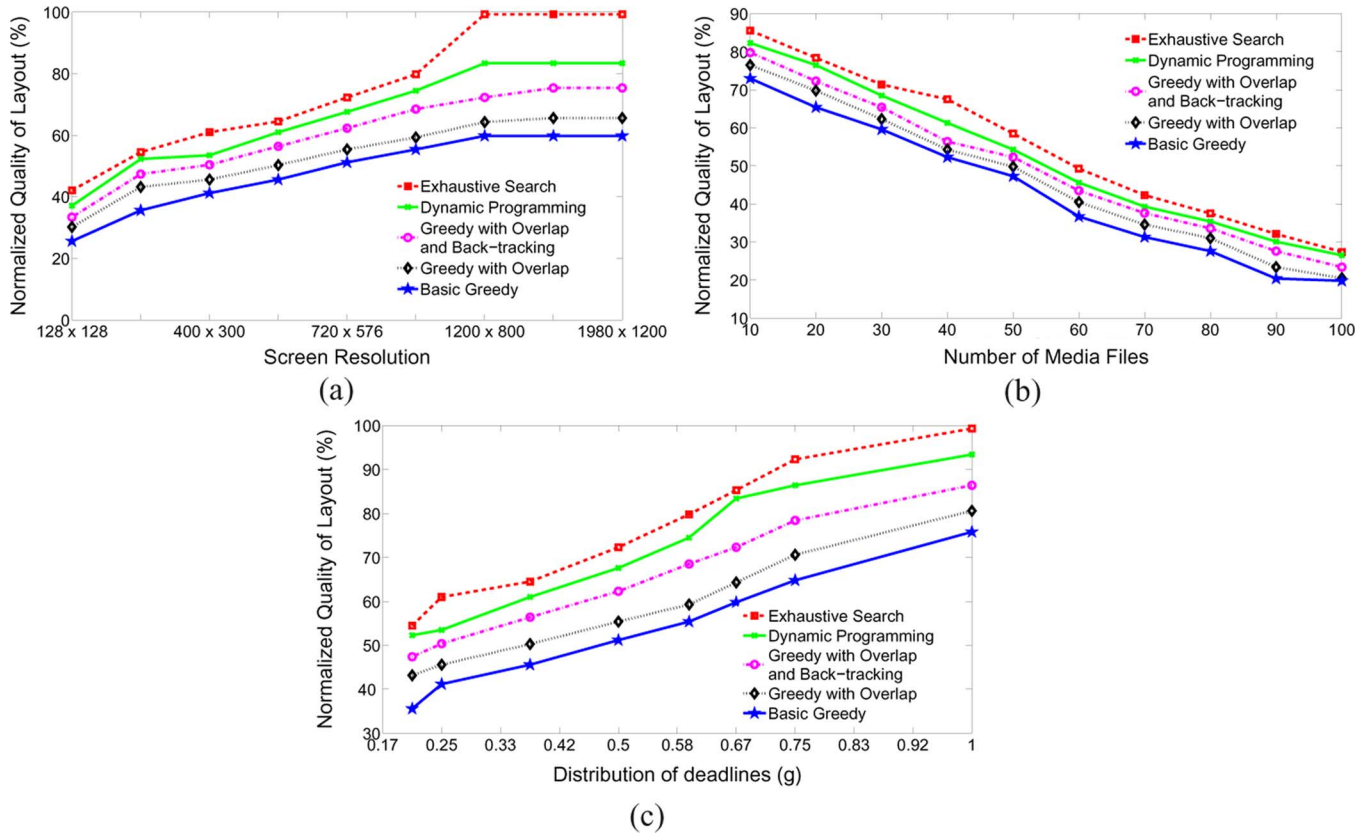

Fig. 9. (a) Normalized Q for different screen resolutions. Q increases at higher resolutions until all the objects are placed and then remains constant. (b) Q decreases with the number of media files because more media files are rejected. (c) Q increase as the numbers of deadlines that occur towards the end of presentations increase. (Higher quality is better. Exhaustive search always produces higher quality layouts.)

contrast, our method includes temporal constraints and the main goal is to maximize the total importance of the information. The performances of existing algorithms depend on how they handle temporal constraints. For a fair comparison, we assume that the packing algorithms are applied at every instant when a new media file is ready to be placed. We use the existing algorithms including one-column heuristic, two-column heuristic, sort by height, sort by width, sort by area, sort by squaredness, and recursive division [33] for a quantitative comparison against our algorithm. We modify the C# source code for these algorithms, available on [33], to consider temporal constraints by applying the packing algorithms every time when a media file is ready to be placed. Other approaches like genetic algorithm and simulated annealing may provide better solutions than our block-based approaches; however, they are computation-intensive and require longer execution time. Hence, we do not use them for comparison.

Table V shows the comparison of existing packing methods with our layout generation methods (first five rows) using metrics such as quality of layouts, screen utilization, and execution time. The values given in this table are obtained by averaging over 100 multimedia presentations, each with 20 media files, when displayed on a screen of resolution 1024×768. These presentations have different media files with different resolutions, preferences, and temporal constraints. Exhaustive search produces the best quality layouts. However, it requires a tremendous amount of time (823 s). Our dynamic programming approach produces layouts whose quality is about 95% (= 76.5/80.2) of the exhaustive search. This approach requires only 17 s to produce the layouts. Greedy approaches produce slightly lower quality layouts; however, they have even lower execution times. Though the other algorithms have higher screen utilization, they produce layouts of poor quality compared with our methods because these algorithms select media files with larger area, but often with lower importance.

5) Practical Issues: Some practical issues that may be encountered in our approach are as follows.

1) The preference types used in our method may not include all the possible relationships among the media files. Hence, the presentation authors can only specify the preference types described in our method, in order to generate TCP-nets. However, it is to be noted that we use the preference types that are used in most of the existing decision-theoretic studies [6]–[8].
2) The total order of importance may not be generated using our method. Two media files having the same number of ancestors and areas cannot be distinguished using our method. Such media files have an equal chance to be chosen. However, when the media files have the same number of ancestors and different areas, we select the media file with the larger area to be more important. The intuition for making this selection is that the selected media file reduces the total amount of unoccupied area. However, there may be situations when selecting the

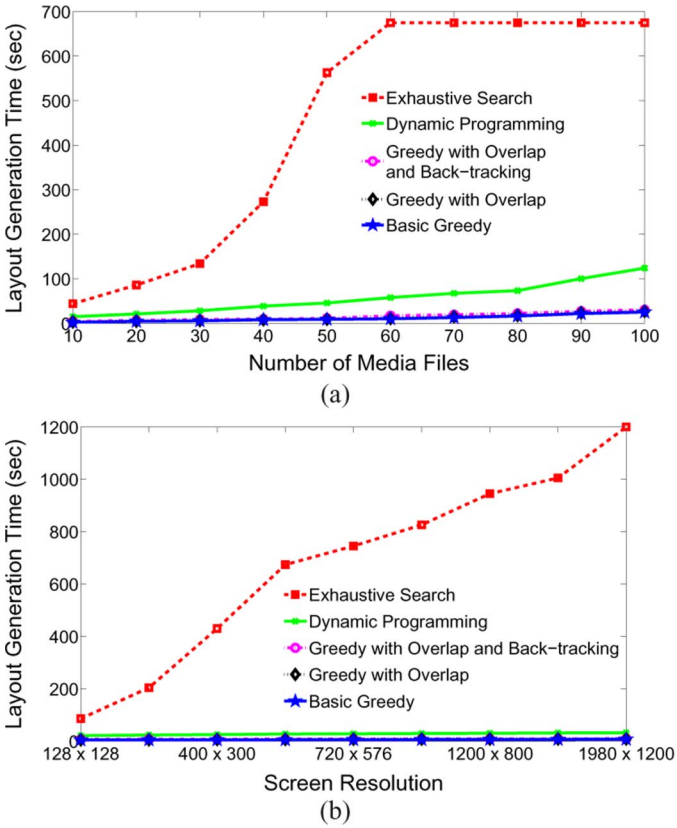

Fig. 10. (a) Layout generation time of various methods for presentations with different numbers of media files. (b) Layout generation time for different screen resolutions.

TABLE V
COMPARISON OF OUR METHODS WITH EXISTING PACKING ALGORITHMS

Technique	Quality (%)	Screen Utilization (%)	Execution Time (secs)
Our methods			
Exhaustive search	80.2	88.7	823
Dynamic programming	76.5	85.6	17.0
Greedy (KAMER)	67.0	80.2	10.0
Greedy with temporal overlap	71.3	81.9	11.1
Greedy with temporal overlap and backtracking	73.2	84.5	12.7
Existing Methods [33]			
One-column heuristic	55.2	83.2	2.1
Two-column heuristic	54.0	81.4	2.3
Sort by height	57.3	92.3	2.4
Sort by width	58.4	93.4	2.4
Sort by Area	59.2	92.7	2.5
Sort by squaredness	56.1	87.5	2.3
Fill by stripes	61.2	93.5	3.4
Recursive Division	62.4	94.4	4.4

media file with smaller area may increase the total number of media files placed on the screen. This may not be the optimal solution and both the approaches may fail for some inputs. The source code is available at https://engineering.purdue.edu/HELPS/opensource.html.

V. CONCLUSION AND FUTURE WORK

We present adaptation algorithms for multimedia presentations displayed on different devices. We propose a method to compute importance of media objects in the presence of preferences and temporal constraints. We present three methods for content adaptation namely exhaustive search, dynamic programming, and greedy algorithms. We also analyze the behavior of these algorithms for various parameters in presentations. This paper may be extended in several ways: 1) Preferences on locations of media files are not considered for layout generation in this paper. The media files are placed on the empty space on the screen. However, the presentation authors may have preferences on where the media files should be placed. 2) This paper considers that all the media files are known *a priori* for generating layouts. However, some presentations may require dynamic addition and removal of media files (for example, breaking news).

REFERENCES

[1] K. Ali, K. Hartmann, G. Fuchs, and H. Schumann, "Adaptive layout for interactive documents," in *Proc. Int. Symp. Smart Graphics*, 2008, pp. 247–255.
[2] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, no. 11, pp. 832–843, Nov. 1983.
[3] K. Bazargan, R. Kaslner, and M. Sarrafzadeh, "Fast template placement for reconfigurable computing systems," *IEEE Des. Test Comput.*, vol. 17, no. 1, pp. 68–83, Jan. 2000.
[4] A. Blekas, J. Garofalakis, and V. Stefanis, "Use of RSS feeds for content adaptation in mobile web browsing," in *Proc. Int. Cross-Disciplinary Workshop Web Access*, 2006, pp. 79–85.
[5] A. Borning, R. K.-H. Lin, and K. Marriott, "Constraint-based document layout for the web," *Multimedia Syst.*, vol. 8, no. 3, pp. 177–189, Oct. 2000.
[6] C. Boutilier, R. I. Brafman, H. H. Hoos, and D. Poole, "Preference-based constrained optimization with CP-nets," *Computat. Intell.*, vol. 20, no. 2, pp. 137–157, Apr. 2004.
[7] R. I. Brafman, C. Domshlak, and S. E. Shimony, "Qualitative decision making in adaptative presentation of structural information," *ACM Trans. Inf. Syst.*, vol. 22, pp. 503–539, Oct. 2004.
[8] R. I. Brafman and D. Friedman, "Adaptive rich media presentations via preference-based constrained optimization," in *Proc. Preferences: Specification, Inference, Applications, Dagstuhl Seminar*, 2006.
[9] M. C. Buchanan and P. T. Zellweger, "Automatic temporal layout mechanisms," in *Proc. ACM Int. Conf. Multimedia*, 1993, pp. 341–350.
[10] M. C. Buchanan and P. T. Zellweger, "Scheduling multimedia documents using temporal constraints," in *Proc. Int. Workshop Network and Operating System Support for Digital Audio and Video*, 1993, pp. 237–249.
[11] V. Cardellini, P. S. Yu, and Y. W. Huang, "Collaborative proxy system for distributed web content transcoding," in *Proc. ACM Int. Conf. Knowledge and Management*, 2000, pp. 520–527.
[12] A. Y. Chang, "Design of an intelligent distributed multimedia presentation using temporal algebra and SMIL," in *Proc. Int. Conf. Multimedia Expo*, 2004, pp. 2211–2214.
[13] S. Edelkamp, S. jabbar, and M. Nazih, "Costoptimal planning with constraints and preferences in large state spaces," in *Proc. Int. Conf. Automated Planning and Scheduling Workshop Preferences and Soft Constraints in Planning*, 2006, pp. 38–45.
[14] J. He, T. Gao, W. Hao, I.-L. Yen, and F. Bastani, "A flexible content adaptation system using a rule-based approach," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 1, pp. 127–140, Jan. 2007.
[15] J.-L. Hsiao, H.-P. Hung, and M.-S. Chen, "Versatile transcoding proxy for internet content adaptation," *IEEE Trans. Multimedia*, vol. 10, no. 4, pp. 646–658, Jun. 2008.
[16] R.-H. Jan, C.-P. Lin, and M.-S. Chern, "An optimization model for web content adaptation," *Int. J. Comput. Telecommun. Netw.*, no. 7, pp. 953–965, May 2006.

[17] M. Jordan, N. Layaida, and T. Vidal, "Using temporal constraints networks to manage temporal scenario of multimedia documents," in *Proc. Eur. Conf. Artificial Intelligence: Workshop Spatial and Temporal Reasoning*, 1998.
[18] C. Jou, "A semantics-based automatic web content adaptation framework for mobile devices," in *Proc. Int. Conf. Web Information Systems and Technologies*, 2007, pp. 230–242.
[19] A. Kinno, H. Yukitomo, and T. Nakayama, "An efficient caching mechanism for XML content adaptation," in *Proc. Int. Multimedia Modeling Conf.*, 2004.
[20] J. Kong, M. Qiu, and K. Zhang, "Authoring multimedia documents through grammatical specifications," in *Proc. Int. Conf. Multimedia and Expo*, 2003, vol. 2, pp. 629–632.
[21] J. Kong and K. Zhang, "Parsing spatial graph grammars," in *Proc. IEEE Symp. Visual Languages—Human Centric Computing*, 2004, pp. 99–101.
[22] J. Kong, K. Zhang, and X. Zeng, "Spatial graph grammars for graphical user interfaces," *ACM Trans. Computer-Human Interact.*, vol. 13, no. 2, pp. 268–307, Jun. 2006.
[23] E. Lee, J. Kang, J. Park, J. Choi, and J. Yang, "Scalable web news adaptation to mobile devices using visual block segmentation for ubiquitous media services," in *Proc. Int. Conf. Multimedia and Ubiquitous Engineering*, 2007, pp. 620–625.
[24] C.-S. Li, R. Mohan, and J. R. Smith, "Multimedia content description in the InfoPyramid," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 1998, vol. 6, pp. 3789–3792.
[25] W. Y. Lum and F. C. M. Lau, "Relationship-aware content adaptation of structured web documents for mobile computing," in *Proc. Int. Conf. Parallel and Distributed Systems*, 2005.
[26] F. B. Megino, J. M. M. Senchez, and V. V. Lopez, "Virtual camera tools for image2video applications," in *Proc. Int. Workshop Image Analysis for Multimedia Interactive Services*, 2008, pp. 0:223–226.
[27] R. Mohan, J. R. Smith, and C.-S. Li, "Adaptive multimedia internet content for universal access," *IEEE Trans. Multimedia*, vol. 1, no. 1, pp. 104–114, Mar. 1999.
[28] K. Nagao, "Constraints and preferences: Integrating grammatical and semantic knowledge for structural disambiguation," in *Proc. Pacific Rim Int. Conf. Artificial Intelligence*, 1990.
[29] Y. Nimmagadda, Source Code for Exhaustive Search, Dynamic Programming, and Greedy Algorithms for Adaptation of Multimedia Presentations. [Online]. Available: https://engineering.purdue.edu/HELPS/opensource.html.
[30] Y. Nimmagadda, K. Kumar, and Y.-H. Lu, "Preference-based adaptation of multimedia presentations for different display sizes," in *Proc. IEEE Int. Conf. Multimedia and Expo*, 2009.
[31] B. Shen, S.-J. Lee, and S. Basu, "Caching strategies in transcoding-enabled proxy systems for streaming media distribution networks," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 375–386, Apr. 2004.
[32] A. J. Solon, P. M. Kevitt, and K. Curran, "TeleMorph: A fuzzy logic approach to network-aware transmoding in mobile intelligent multimedia presentation systems," *IEEE J. Select. Topics Signal Process.*, vol. 1, no. 2, pp. 254–263, Aug. 2007.
[33] R. Stephens, Source Code for Packing Algorithms. [Online]. Available: http://www.devx.com/dotnet/Article/36005.
[34] X. Tang, F. Zhang, and S. T. Chanson, "Streaming media caching algorithms for transcoding proxies," in *Proc. Int. Conf. Parallel Processing*, 2002.
[35] T. C. Thang, Y. J. Jung, and Y. M. Ro, "Modality conversion for QoS management in universal multimedia access," *Proc. Inst. Elect. Eng., Vis., Image, Signal Process.*, vol. 152, pp. 374–384, 2005.
[36] S. J. H. Yang, I. Y. L. Chen, and R. Chen, "Applying content adaptation to mobile learning," in *Proc. Int. Conf. Innovative Computing Information and Control*, 2007.

Yamini Nimmagadda (S'07) received the B.Tech. degree from the Indian Institute of Technology, Guwahati. She is pursuing the Ph.D. degree in the School of Electrical and Computer Engineering at Purdue University, West Lafayette, IN.

Her research interests include resource management for heterogeneous devices and mobile multimedia applications.

Karthik Kumar (S'09) received the M.S. degree from Purdue University, West Lafayette, IN, where he is pursuing the Ph.D. degree in the School of Electrical and Computer Engineering.

His research interests include energy conservation for computer systems.

Yung-Hsiang Lu (SM'08) received the B.S.E.E. degree from National Taiwan University, Taipei, and the Ph.D. degree from Stanford University, Stanford, CA.

He is an Associate Professor in the School of Electrical and Computer Engineering of Purdue University, West Lafayette, IN. His research focuses on system-level energy conservation and resource management.

Prof. Lu was one of the three recipients of Purdue "Class of 1922 Helping Student Learn Award" in 2008. In 2005, he received the Purdue ECE Chicago Alumni Award. In 2004, he obtained a Career Award from the NSF National Science Foundation. He is a senior member of the ACM.