Improving the Dynamic Behavior of CP-net based MultiMedia Systems by Predicting Likely Components

Carmel Domshlak Solomon E. Shimony
Department of Computer Science
Ben-Gurion University of the Negev
Beer-Sheva 84105, Israel
{dcarmel,shimony}@cs.bgu.ac.il

Abstract

We present a scheme for real-time dynamic presentation of multi-media documents. The system uses CP-nets to describe the author's presentation preferences, and a partial instantiation of component configuration variables to denote viewer choices. The system presents components by finding the optimal assignment in the CP-net constrained by the user choices.

Performance is important, and can be enhanced by predicting components needed in the near future, as well as their form. We assume that a viewer event distribution is given, reducing the prediction problem to that of finding marginal probabilities over components. For the single-step prediction problem, we present a low-order (quadratic) polynomial-time algorithm. Multi-step prediction, is also an easy problem, provided one can compute the distribution over events efficiently.

Introduction

Acquiring and presenting information are knowledge-intensive activities that in combination have come to be known as authoring. The task of an author is to collect a coherent body of information into a document, structure it in a meaningful way, and present it in an appropriate manner to a set of viewers. This traditional notion of authoring commits the author to the form and the content of the document, well in advance of the actual presentation time. The author must both select and order the information to be presented.

An important goal of modern document authoring and presentation systems is to provide viewer-oriented personalization of document content, since presentation can have a significant impact on how well the information is communicated to the viewer. This goals is of particular importance given the growing volumes of multimedia content, and has led to great interest in intelligent multi-media presentation systems both in industry and academia. These systems exploit their knowledge base in order to dynamically adapt and draw design decisions according to the runtime requirements of user-computer interaction (Bordegoni *et al.* 1998; Csinger, Booth, & Poole 1995; Karagiannidis, Koumpis, & Stephanidis 1998; Roth & Hefley 1993).

A conceptually new model for representing a multimedia document content is proposed in (Domshlak, Brafman, & Shimony 2001) and illustrated on a prototype system for web-page authoring and presentation. This model is unique in two ways. First, it emphasizes the role of the author in the process, viewing her as a content expert whose knowledge and taste are important factors in how the document will be presented. The resulting model exhibits dynamic response to user preferences, but does not require learning long-term user profiles. Second, to accomplish this behavior, well-founded tools for preferences elicitation and preference optimization are used – namely, *conditional preference networks* (CP-nets, for short). These tools are grounded in qualitative decision theory (Doyle & Thomason 1999), and help the author structure her preferences over document content off-line, in an intuitive manner, and support fast algorithms for optimal configuration determination.

The move to personalized, context dependent dynamic presentation of multimedia documents – regardless of the particular approach to personalization – raises a serious performance issue. Large amounts of information must be delivered to the user quickly, on demand. To see the issues involved, consider a medical record in which (multimedia) patient information is continuously gathered. It arrives from different clinics, diagnostic centers, homes, laboratories, etc. In addition, some components have several presentation options. Some presentation options may have to be pre-generated and stored to be later delivered on demand. For example, a CT image can be presented either plain, or segmented. If online, high quality segmentation is infeasible, it must be performed offline, stored, and then transmitted on demand. A medical record may be accessed remotely from a web-site, as in the case of web pages in (Domshlak, Brafman, & Shimony 2001), or by other means from a centralized database serving a number of physically distant clinics. In all such cases, the physician viewing it should be provided with the quickest possible response time to her choices. Two related problems hamper our ability to provide fast response times to online, user dependent, presentation needs:

1. The buffer available on the client side may be much smaller than the size of the requested multimedia document.

2. Bandwidth limitations are likely to make downloading even a single document component upon request a lengthy process.

Thus, when tackling the presentation of media-rich information, in addition to the decision *what to present* from the whole content of the document (which we address using the model from (Domshlak, Brafman, & Shimony 2001)), one must also address the question of *how to ensure that this presentation is done in a timely fashion*. Note again that this problem is not unique to our particular approach to dynamic document presentation. In fact, a problem of similar flavor, web-caching, has received much attention in industry and academia (Wang 1999).

One attractive solution, reminiscent of the idea of caching, is that of pre-fetching likely components ahead of time. Ideally, we would have liked to download the whole document ahead of time. However, the limited buffer size and communication bandwidth prevent this. Instead, we download the components that are most likely to be requested by the user, using the user's buffer as a cache. Thus, in this paper we extend the model for CP-net based multimedia systems (Domshlak, Brafman, & Shimony 2001) by a *preference-based optimized prefetching* of the document components. Indeed, we believe that the low computational complexity of our prefetch algorithms provides another supporting argument for the use of this technique.

Our optimized prefetch approach works as follows: We assume a given distribution over possible user actions, and for each document presentation mode of each document component, we compute its likelihood of being a part of the document's next configuration. Using this information, we can determine *what to deploy* to the buffer on the client side, and *what to remove (suppress)* from this buffer. Somewhat surprisingly, the CP-net semantics allows us to determine this information in time which is a low order polynomial in the number of components. This is important as prediction should be made in real-time, in order to be useful.

We note in passing that much work exists on improving performance of caches, (by attempting to predict future usage of memory pages, disk blocks, web pages, etc.) in the (mostly operating systems related) research literature. However, the problem defined here, in the context of the constraints imposed by the particular qualitative decision model (CP-nets), is completely unexplored, and the above research is thus largely irrelevant.

In the next section we provide the necessary background on CP-net based multimedia systems. The following section formally defines the prediction problem, and provides a solution for single-step prediction. We conclude with a discussion of some aspects of multi-step prediction.

Background

This section provides background on CP-net based multimedia systems, originally introduced in (Domshlak, Brafman, & Shimony 2001). In particular we discuss qualitative preferential statements, the CP-net model, and the overall architecture of the CP-net based multimedia system.

Qualitative Preferences and CP-nets

Any multimedia document can be considered as a set of components $\{C_1, \ldots, C_n\}$. Each component is associated with its content. For example, the content of a component may be a block of text, an image, etc. Each component may have several optional presentation modes to the viewer, and these options for C_i are denoted by $D(C_i) = \{c_{i,1}, \ldots, c_{i,i_m}\}$. For example, CT image in a medical record can be presented in the flat form, in the segmented form, or omitted altogether.

The document's components define a configuration space $\mathcal{C} = D(C_i), \times \cdots \times D(C_n)$. Each element σ in this space is a possible presentation (= configuration) of the document content. Our task will be to determine the preferentially optimal presentation, and to present it to the current viewers of the document. In terms of decision theory, the set of components of a document is a set of features, the optional presentations of a document's content are the values of the corresponding features, and presentations are outcomes, over which a preference ranking can be defined.

A preference order $\succeq$ over the configuration space is defined as follows: $\sigma_1 \succeq \sigma_2$ means that the decision maker views configuration σ_1 as equal to or more preferred than σ_2. This preference ranking is a partial order, and, of course, it will be different for different decision makers. Given a preference order $\succeq$ over the configuration space, an *optimal configuration* is any $\sigma \in \mathcal{C}$ such that $\sigma \succeq \sigma'$ for any $\sigma' \in \mathcal{C}$.

The preference order reflects the preferences of a decision maker. The typical decision maker in preference-based product configuration is the consumer. However, in our application the role of the decision maker is relegated to another actor – the document authors. The authors are the content experts, and they are likely to have considerable knowledge about appropriate content presentation. We would like the document to reflect their expertise as much as possible.

During the, possibly continuous, creation of the document, the authors describe their expectations regarding content presentation. Therefore, the preference order $\succeq$ represents the static subjective preferences of the document authors, not of its viewer. Thus, preference elicitation is performed on the document authors off-line once for all subsequent accesses to the created document. The dynamic nature of the document presentation stems from the interaction between the statically defined author preferences and the constantly changing content constraints imposed by recent choices of the current viewers.

These requirements are addressed in the CP-net based framework for preference-based multimedia documents presentation, introduced in (Domshlak, Brafman, & Shimony 2001). The CP-net model (Boutilier *et al.* 1999) is an intuitive, qualitative, graphical model, that represents statements of conditional preference under a *ceteris paribus* (all else equal) assumption. In terms of our domain, this conditional *ceteris paribus* semantics requires the authors of the multimedia document to specify, for any specific component C_i of interest, the content presentation of which other components $\Pi(C_i)$ can impact her preferences over the presentation options of C_i. For each content configuration A of $\Pi(C_i)$, the

author must specify her preference ordering over the presentation options of C_i given A.

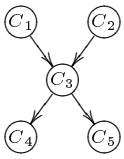
For example, let C_i be a component with domain $D = \{c_{i,1}, c_{i,2}\}$, and suppose that an author determines that $\Pi(C_i) = \{C_j, C_k\}$ and that $c_{i,1}$ is preferred to $c_{i,2}$ given that C_j appears as $c_{j,x}$ and C_k appears as $c_{k,y}$, *all else being equal*. This means that given two configurations that agree on all components other than C_i and in which $C_j = c_{j,x}$ and $C_k = c_{k,y}$, the configuration in which $C_i = c_{i,1}$ is preferred to the configuration in which $C_i = c_{i,2}$.

CP-nets bear a surface similarity to Bayesian networks. Both rely employ annotated directed graphs in which nodes correspond to features. In our particular application, each feature represents the viewing mode of a particular component C_i of the document. The immediate ancestors of C_i in the graph are associated with $\Pi(C_i)$. Formally, if $\overline{C_i} = \{C_1, \ldots, C_n\} \setminus \{C_i, \Pi(C_i)\}$ then C_i and $\overline{C_i}$ are *conditionally preferentially independent* given $\Pi(C_i)$. This standard notion of multi-attribute utility theory can be defined as follows: Let X, Y, and Z be non-empty sets that form a partition of feature set F. X and Y are conditionally preferentially independent given Z, if for each assignment z on Z and for each x_1, x_2, y_1, y_2 we have that

$$x_1 y_1 z \succeq x_2 y_1 z \text{ iff } x_1 y_2 z \succeq x_2 y_2 z.$$

Each node of the CP-net contains a table, denoted by $CPT(C_i)$ (*conditional preference table*), which describes the preferences about the values of the corresponding feature C_i given all possible combinations of $\Pi(C_i)$.

An example CP-net with the corresponding preference table is shown in Figure 1. We see that the author specifies unconditional preference for presenting the content of component C_1 (denoted in figure by $c_{1,1} \succ c_{1,2}$). However, if C_1 is presented by $c_{1,1}$ and C_2 is presented by $c_{2,2}$, then the author prefers to present the content of C_3 by $c_{3,2}$ (denoted by $(c_{1,1} \wedge c_{2,2}) : c_{3,2} \succ c_{3,1}$).



$\mathbf{C_i}$	$\mathbf{CPT(C_i)}$
C_1	$[c_{1,1} \succ c_{1,2}]$
C_2	$[c_{2,2} \succ c_{2,1}]$
C_3	$[(c_{1,1} \wedge c_{2,1}) \vee (c_{1,2} \wedge c_{2,2}) : c_{3,1} \succ c_{3,2};$ $(c_{1,1} \wedge c_{2,2}) \vee (c_{1,2} \wedge c_{2,1}) : c_{3,2} \succ c_{3,1}]$
C_4	$[c_{3,1} : c_{4,1} \succ c_{4,2};\ c_{3,2} : c_{4,2} \succ c_{4,1}]$
C_5	$[c_{3,1} : c_{5,1} \succ c_{5,2};\ c_{3,2} : c_{5,2} \succ c_{5,1}]$

Figure 1: An example CP-net

One of the central properties of the CP-net model is that, given a CP-net $\mathcal{N}$, one can easily determine the preferentially optimal outcome (Boutilier *et al.* 1999): Traverse the nodes of $\mathcal{N}$ according to a topological ordering and set the value of the processed node to its preferred value, given the (already fixed) values of its parents. Indeed, any CP-net determines a unique best outcome. More generally, suppose that we are given "evidence" constraining outcomes in the form of a partial assignment π on the variables of $\mathcal{N}$. Determining the best completion of π, i.e., the best outcome consistent with π, can be achieved in a similar fashion by projecting π on the corresponding variables in $\mathcal{N}$ before the top-down traversal described above. In what follows, we refer to this procedure as **Algorithm 1**.

CP-net based Multimedia System Architecture

A CP-net based multimedia system consist of two tools - the *authoring tool*, and the *viewing tool*. The central part of the authoring tool is a module for specification of the CP-net for the created/edited multimedia document. The result of such preference-based multimedia document design, is a document specifying both what to present and how to present. Given such a document, the viewing tool is responsible for reasoning about the preferences, i.e. for an optimal content reconfiguration after an interaction of the viewer with the document. In this process, the user's recent content choices are viewed as constraints – these items must appear in the specified manner – subject to which an optimal document presentation with respect to the author's CP-net must be generated. The outline of the system is presented in Figure 2.

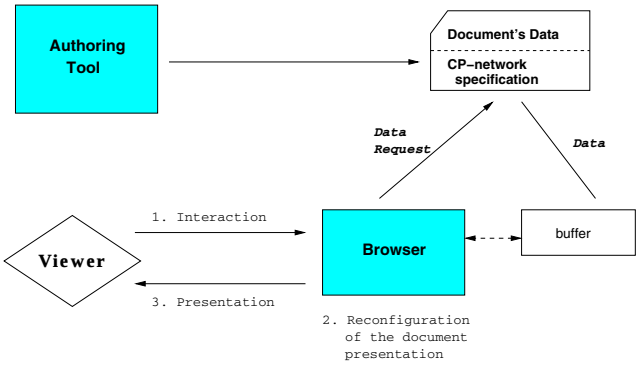


Figure 2: CP-net based multimedia system framework

The viewing tool on the client side is a specialized browser: Accessing a CP-net based multimedia document results in shipping the CP-net, and the data of the document to the browser's buffer. The amount of the information that is actually shipped is subject to the size of the buffer. If some data is subsequently required for document presentation, and it is not available in the buffer, it will be shipped from the server side on demand. The general algorithm for such a browser is presented in Figure 3. Upon downloading the document, the agent sets all the components to their values (= content) in the optimal presentation, given the prior model $\mathcal{M}$ of the viewer's current interests. Note that the optimal presentation for many component may be simply to hide them completely, e.g., due to their current irrelevance. The representation of the model $\mathcal{M}$, as well as the method for acquiring the model (or learning it), are system specific. However, any model $\mathcal{M}$ that explicitly specifies the form of content presentation for some of the document components (i.e. constitutes a partial assignment of values to the C_i's) can be used with the general framework.

Steps 1 and 2 of the Optimized-Presentation algorithm stand for Algorithm 1. Note that if some data to be presented is not present in the local buffer, it is shipped online from the server side (step 3). Now, since the browser is responsible for reacting to viewer actions by reconfiguring the content presentation of the document, henceforth it acts in event-driven fashion, and waits for a viewer's interaction with the

Procedure Optimized-Presentation

$\mathcal{N}$ - CP-net of the multimedia document's components that describes the preferences of the author.

$\mathcal{M}$ - User model - an assignment to some of the C_i's

loop:

1. Set the values of the components in $\mathcal{N}$ addressed by $\mathcal{M}$ to the values specified by $\mathcal{M}$.
2. Traverse $\mathcal{N}$ in a topological order and set each unspecified component C_i to its most preferred value $c_{i,j}$ w.r.t. to the values of its predecessors in $\mathcal{N}$.
3. For each component value $(C_i, c_{i,j})$ from steps 1- 2, not available in the local buffer, download it into the local buffer.
4. Complete the presentation, and wait for user feedback.
5. Update $\mathcal{M}$ w.r.t. the received feedback.

Figure 3: Preference-based reconfiguration algorithm

browser (step 4). After some feedback from the viewer is received, the user model $\mathcal{M}$ is updated (step 5), and reconfiguration of the document presentation is performed. The new content presentation will be optimal with respect to the current user model $\mathcal{M}$, and the CP-net of the document.

Predicting most preferred outcomes

To improve the response time to dynamic changes in the document's presentation, we attempt to predict which components have a high probability of being required soon. Memory usage can be optimized by discarding components that are not likely to be useful in the near future and replacing them with those that are likely to be required. Below we show how the probability that a component will be needed can be computed efficiently. But first, we provide an overview of the process and define some needed notation.

The Framework

Recall that changes to document presentation are caused by user actions, e.g. following a link, and their effect depends on the particular design choices made. For example, in (Domshlak, Brafman, & Shimony 2001), user actions are limited to requests to display certain items. A list of the last k requests is maintained, and the optimal set of components containing these k requests is displayed. More generally, at each point the user places certain constraints on the presentation (e.g., "make sure I see these k components") and an optimized presentation based on these constraints is selected. An important assumption of our model is that some statistics on events is available. For example, this is the case in e-commerce web-sites, most of which maintain user logs.

We model user constraints simply as partial assignments of presentation choices to different components. That is, the user may indicate that for component C_i a presentation choice of c_i is required. This assignment is partial because on the value of some components, the user places no constraints. Thus, when describing user presentation constraints, it is convenient to think of $D_e(C_i)$, the *extended* domain of each variable (i.e., component) C_i. $D_e(C_i)$ is defined as $D(C_i) \cup \{\mathsf{u}\}$, where $\{\mathsf{u}\}$ denotes the fact that C_i is unselected, i.e., no constraint is placed on its presentation. We use A^t to denote the *selection state* of the set of components at time t. A^t is an element of $D_e(C_1) \times \cdots \times D_e(C_n)$.

At various points in time, the user indicates a change of preference. We refer to these changes as *events* and denote the set of possible events by E. An event may consist of adding a new constraint (i.e., selecting a presentation mode for a component), or relaxing the current constraints by unselecting a component (i.e., assigning the value $\{\mathsf{u}\}$ to it). An event changes the current selection state.

Because the precise manner in which a new event changes the new selection state may depend on the identity of previous events, we define the notion of a *preference history state* at time t, or *preference state* for short. The preference state at time t, S^t, consists of the current selection state together with the list of all previous events. Thus, $S^t = (A^t, E^0, E^1, \ldots, E^t)$. We denote the set of all such states by S. Finally, the *transition function* F determines how the selection state (and thus, the preference state) changes as a result of a new event. That is, $F : S \times E \rightarrow S$.

The Prediction Problem

The prediction problem is to determine the distribution over the next optimal document configuration. We assume that we have a model of $P(E^{t+1}|S^t)$ – note how the dependence on previous events enters through the inclusion of previous events in S^t – and we need to compute the marginal probabilities of the different component configurations after one or more future events.

Denoting the fact that component C_i has the value $c_{i,j}$ at time t by $(C_i^t = c_{i,j})$, we need to compute $P(C_i^{T+l} = c_{i,j}|S^T)$ for $l = 1, 2, ...$, and $c_{i,j} \in D(C_i)$.[1] We begin here with the problem of predicting the most-likely configuration of all components *one* time-step ahead (i.e. only $l = 1$), and discuss the more general prediction problem later on.

For the single-step prediction problem (defined below), we only need the distribution for the next event $P(E^{T+1}|S^T)$, which is directly available in the user model.

Definition 1 *The single-step prediction problem is: given a CP-net, a transition function F, a state S^T, and a distribution $P(E^{T+1}|S^T)$, find $P(C_i^{T+1}|S^T)$ for $1 \leq i \leq n$.*

In order to use our prediction model in practice, we need to further specify the transition function F. We consider two types of transition functions:

scheme 1: A^{t+1} depends only on A^t and E^{t+1} as follows: the event E^{t+1} is an assignment to some C_i. Make A^{t+1} the same as A^t, except change the assignment to C_i to be as specified by E^{t+1}.

scheme 2: A^{t+1} depends only on the k last events (from $E^0, E^1, ..., E^t, E^{t+1}$), as follows. A^{t+1} consists only of assignments made in events $E^{t-k+2}, ..., E^t, E^{t+1}$. If several assignments are made to the same variable, the latest

[1] The standard shorthand, where for any variable X, $P(X)$ stands for the set of probabilities $\{P(X = x)|x \in D(X)\}$, is used throughout. Thus, we denote the set of values to compute by $P(C_i^{T+l}|S^T)$ (for $l = 1, 2, ...$).

assignment overrides earlier assignments. All variables that are not mentioned in the last k events are assigned the value u, i.e., no constraint is placed on them.

The second scheme corresponds to the treatment of user preferences in earlier work on web page configuration (Domshlak, Brafman, & Shimony 2001), where k was called the *event queue length* and was one of the properties of a configurable web page). The first scheme is reasonable as well. Our treatment below covers both cases.

We begin with the intuitively obvious conditioning:

$$P(C_i^{T+1}|S^T) = \sum_{A^{T+1}} P(A^{T+1}|S^T)P(C_i^{T+1}|A^{T+1}, S^T) \quad (1)$$

where A^{T+1} ranges over all possible selection states. In practice, we need only sum over a small number of selection states reachable from S^T via a single event since for unreachable states, $P(A^{T+1}|S^T) = 0$. The second multiplicative term, $P(C_i^{T+1}|A^{T+1}, S^T)$ depends only on A^{T+1}. This is a degenerate distribution: $P(C_i^{T+1} = c_{i,j}|A^{T+1})$ is 1 just when $c_{i,j}$ is in the most preferred outcome in the CP-net given A^{T+1}, and 0 otherwise. The value of $P(C_i^{T+1}|A^{T+1})$ for a given A^{T+1} can be computed with a single (linear time) pass over the CP-net (Algorithm 1), for all the marginals $C_i = c_{i,j}$ with $0 \leq i \leq n$, and $c_{i,j} \in D(C_i)$.

The first multiplicative term in Eq. 1 can be computed from transition function F and the event distribution:

$$P(A^{T+1}|S^T) = \sum_{F(S^T,E^{T+1})=S^{T+1}} P(E^{T+1}|S^T)$$

Algorithm 2
Input: CP-net $\mathcal{N}$, assignment A^T, discrete distribution $P(E_{i,j})$ over $(1 \leq i \leq n, D_e(C_i))$
Output: $P(C_i^{T+1}|A^T)$ for all C_i in the CP-net.

1. Initialization: set variables $P_{i,j} = 0$, one for each $1 \leq i \leq n$, and each $1 \leq j \leq |D(C_i)|$
2. **for** $i = 1$ **to** n **do**
3. **for** $j = 1$ **to** $|D_e(C_l)|$ **do**
4. Set A to A^T modified by assignment $C_i = c_{i,j}$
5. ProbCalculation $(\mathcal{N}, A, P(E_{i,j}))$.
6. **return** the values $P_{i,j}$ as $P(C_i^{T+1} = c_{i,j}|A^T)$, respectively.

ProbCalculation $(\mathcal{N}, A, P(E_{i,j}))$
1. Determine preferentially optimal assignment α for $\mathcal{N}$, consistent with A, according to algorithm 1.
2. **for** $l = 1$ **to** n **do**
3. **if** $\alpha[l] = c_{l,m}$ **then**
4. $P_{l,m} = P_{l,m} + P(E_{i,j})$

Figure 4: Single-step prediction for scheme 1

In general, this computation may be hard. However, in both schemes considered here for the transition function F, the summation will be over a small number of possible events, and additionally most states S^{T+1} will not be reachable from S^T by a single event E^{T+1}. In particular:

1. For F as defined in scheme 1, there are only $O(nd)$ possible reachable states, since an event causes a change to A^t in only *one* variable, at most.
2. For F as defined in scheme 2, there are only $O(nd)$ possible reachable states as well, since an event assigns a value to at most one more variable. Here, the oldest event drops off the queue, thus a new event also causes the value of one variable C_i to become u, i.e., unconstrained. This change is independent of which event occurs - unless the event consists of an assignment to C_i, but this latter exception does not contribute to the asymptotic complexity.

Theorem 1 *Given a CP-net with n variables, and maximum domain size d, complexity of single-step prediction under transition function F in schemes 1 or 2, is $O(n^2d)$.*

The proof is constructive - the algorithms are presented in what follows. Notice that in our framework the queue size k is always less than n.

Algorithm 3
Input: CP-net $\mathcal{N}$, $k-1$ recent events $\langle E^{t-k+2}, \ldots, E^t \rangle$, discrete distribution $P(E_{i,j})$ over $(1 \leq i \leq n, D_e(C_i))$.
Output: Probability distribution $P(C_i^{t+1}|S^t)$

0. Create a set of variables $P_{i,j} = 0$, one for each $1 \leq i \leq n$, and each $1 \leq j \leq |D(C_i)|$
1. Create a template assignment A^t on the variables of $\mathcal{N}$:
2. **for** $i = 1$ **to** n **do** $A^t(i) =$ u
3. **for** $j = t - k + 2$ **to** t **do**
4. **if** $E^j = (C_i, c_{i,j})$ **then** $A^t(i) = c_{i,j}$
5. **let** $V \subset C$ be the set of variables affected by $E^{t-k+2}, \ldots, E^t$.
6. **foreach** $C_i \in C$ **do**
7. **if** $C_i \in V$ **then** $D = D_e(C_i)$ **else** $D = D(C_i)$
8. **for** $j = 1$ **to** $|D_e(C_i)|$ **do**
9. $A = A^t$ **and** $A(i) = c_{i,j}$
10. ProbCalculation $(\mathcal{N}, A, P(E_{i,j}))$.
11. **return** the values $P_{i,j}$ as $P(C_i^{T+1} = c_{i,j}|\langle E^{t-k+2}, \ldots, E^t \rangle)$, respectively.

Figure 5: Single-step prediction for scheme 2

We begin with the algorithm for scheme 1, where we essentially ignore past events and consider only A^T. The algorithm is presented in Figure 4, and its central procedure is denoted by Algorithm 2.

For each possible event (lines 2-3) we proceed as follows: First, we generate the corresponding next user selection state A (line 4) by updating the current user selection state A^T by the (variable, value) induced by the event. Then, in line 5, we pass the resulting user selection state A to the ProbCalculation procedure that

1. generates the optimal outcome α consistent with A using Algorithm 1, and
2. if $c_{l,m}$ is the value of C_l induced by the optimal configuration α, updates $P_{l,m}$ (accumulating $P(C_l^{T+1} = c_{l,m}|A^T)$) by the probability of event $E_{i,j}$.

Lines 4-5 are performed for each possible event, thus are executed $O(nd)$ times. Complexity of ProbCalculation is $O(n)$, resulting in overall complexity of $O(n^2d)$.

Figure 5 presents the algorithm for scheme 2, where we consider only k past events $\langle E^{t-k+2}, \ldots, E^t \rangle$. First, in lines 1-4, we translate the recent events into a current user selection state A^t, and this is done in $O(n)$.

The complexity of lines 5-10 is $O(n^2d)$ as in Algorithm 2. However, in certain cases we can exploit the fact that the $k << n$. Lines 7 distinguishes between the events involving values specifies by A (at most kd such events), and those not specified by A. Total complexity of probability updating for the former set of events is $O(nkd)$, and for the latter set is $O(n^2d)$. A specialized, linear-time algorithm for the latter set exists for singly connected CP-nets (discussed in the full version of the paper), reducing the overall complexity to $O(nd(k+d))$. This is essentially linear time, since in a typical application k and d are bounded by a small constant.

Multi-Step Prediction Problem

It is frequently desirable to perform multi-step prediction, i.e. to compute $P(C_i^{T+t})$ for some $t > 1$. In order to do that, we first need to model the dependencies between the events $E^{T+1}, E^{T+2}, \ldots$. We briefly consider three possibilities:

1. The events are independent given the state at time T.
2. Event E^{T+t} depends on A^T and on some previous events, E^{T+l}, where $1 \leq l < t$. This is the most general case.
3. The event E^{T+t} depends only on the configuration at time $T+t-1$. Note that we do not rule out dependency on previous events, but here, dependency must be summarized by the configuration.

In all cases, there are potentially $O((nd)^t)$ possible reachable states at time step $T+t$, and if t is very small (e.g., 1 or 2), we can modify our original selection algorithm to handle this problem efficiently. Dependencies among the events can be reasonably modeled as a Bayes network. Of particular interest is Scheme 2, but with a very short event queue; especially when the event queue size is 1, as used in (Domshlak, Brafman, & Shimony 2001). In this case, the number of reachable states is small. We examine the case of a single-element queue, under the above three event dependency models. An important point to note here is that the component configuration in this case does not depend on the selection state, but only on the last event.

First, consider the case of independent events. Now, the multi-step prediction problem is the same as the single-step prediction problem, since neither the state nor the event depend on history. Next, consider the more difficult case of dependent events. The dependence here is restricted to previous events. The complexity of prediction is determined by the complexity of finding the marginal distribution of E^{T+t}, which can be exponential in t in the worst case. After finding the marginals, the multi-step prediction problem is reduced, in this case, to the single-step prediction problem.

The most interesting case is where the next event depends only on the current component configuration. This assumption makes intuitive sense, as the component configuration is observable by the user, and hence most likely to directly influence the next user choice. In this case, we can essentially use the CP-net as an element of a temporal Bayes network and use this network to perform prediction. If the description of the conditional probability of the next event given the configuration is compact, this can be done efficiently, even if the event depends on many (or all) configuration variables. This is due to the fact that there is only a small (linear in n) number of possible optimal configurations – at most one for each possible selection. The result can be stated as follows:

Theorem 2 *For a queue size of 1, the complexity of predicting the marginals over the configuration t time-steps is $O(tn^2d^2)$.*

Proof: By induction on t. At each step we compute the marginals over the configuration, and the distribution over the next event, in time $O(n^2d^2)$. The algorithm is as follows.

1. Initialize accumulators $e_{i,j}$ and $p_{i,j}$ to 0, for all $1 \leq i \leq n$ and $1 \leq j \leq |D_e(C_i)|$.
2. **foreach** possible event E^{T+t} **do**
   (a) Compute the most preferred configuration A.
   (b) **foreach** $1 \leq i \leq n$ and j such that $A(C_i) = c_{i,j}$, let $p_{i,j} = p_{i,j} + P(E^{T+t})$.
   (c) **foreach** $1 \leq l \leq n$ and $1 \leq m \leq |D_e(C_i)|$, let $e_{l,m} = e_{l,m} + P(E^{T+t})P(E^{T+t+1} = E_{l,m}|A)$.
   (d) Output all $p_{i,j}$ as the marginals over the configuration, and $e_{i,j}$ as the distribution over E^{T+t+1}.

Conclusion

To support reasonable quality of service levels during multimedia document presentation on devices with limited buffers and bandwidth limitation, some form of component caching, or prefetching is required. In this paper, we showed that an efficient prefetching algorithm can be provided for the CP-net based presentation approach of (Domshlak, Brafman, & Shimony 2001). This algorithm utilizes the special properties of the underlying CP-net and makes the case for their use in this context even more attractive. The algorithm is used in our application to predict likely multimedia components in real time. In the longer version of this paper, we show how the quadratic time algorithm featured in this paper can be replaced by a more sophisticated, linear time algorithm.

Integration of the above prediction scheme as a module within a system for multimedia conferencing being developed at our institution is underway. Although we expect performance improvements as a result, it is extremely difficult to measure quantitatively in the real system, due to system variability and other uncontrollable factors. Thus, in order to quantify performance enhancement, we intend to construct and experiment on an environment that simulates user choices and communication delays. Simulated performance using our scheme will be compared with, on the one hand a system that does no prefetching, and on the other hand a system that uses a standard prefetch scheme, based on prior marginal distribution over components.

Acknowledgments

Partially supported by an infrastructure grant from the Israeli Science Ministry, and the Paul Ivanier Center for Robotics and Production Management, Ben-Gurion University.

References

Bordegoni, M.; Faconti, G.; Maybury, M.; Rist, T.; Ruggieri, S.; Trahanias, P.; and Wilson, M. 1998. A Standard Reference Model for Intelligent Multimedia Presentation Systems. *Computer Standards and Interfaces* 18(6-7):477–496.

Boutilier, C.; Brafman, R.; Hoos, H.; and Poole, D. 1999. Reasoning with Conditional Ceteris Paribus Preference Statements. In *Proc. of UAI-99*, 71–80.

Csinger, A.; Booth, K. S.; and Poole, D. 1995. AI Meets Authoring: User Models for Intelligent Multimedia. *Artificial Intelligence Review* 8:447–468.

Domshlak, C.; Brafman, R.; and Shimony, S. E. 2001. Preference-based Configuration of Web Page Content. In *Proc. of IJCAI-01*, 1451–1456.

Doyle, J., and Thomason, R. 1999. Background to Qualitative Decision Theory. *AI Magazine* 20(2):55–68.

Karagiannidis, C.; Koumpis, A.; and Stephanidis, C. 1998. Adaption in IMMPS as a Decision Making Process. *Computer Standards and Interfaces* 18(6-7).

Roth, S., and Hefley, W. 1993. Intelligent Multimedia Presentation Systems: Research and Principles. In *Intelligent Multimedia Interfaces*. AAAI Press. 13–58.

Wang, J. 1999. A Survey of Web Caching Schemes for the Internet. *ACM Computer Communication Review* 29(5):36–46.