

Preference-Based Configuration of Web Page Content*

Carmel Domshlak Ronen I. Brafman Solomon E. Shimony

Department of Computer Science
Ben-Gurion University of the Negev
Beer-Sheva 84105, Israel

{dcarmel,brafman,shimony}@cs.bgu.ac.il

Abstract

We present a new approach for personalized presentation of web-page content. This approach is based on preference-based constrained optimization techniques rooted in qualitative decision-theory. In our approach, web-page personalization is viewed as a configuration problem whose goal is to determine the optimal presentation of a web-page while taking into account the preferences of the web author, layout constraints, and viewer interaction with the browser. The preferences of the web-page author are represented by a CP-network, a graphical, qualitative preference model developed in [Boutilier *et al.*, 1999]. The layout constraints are represented as geometric constraints. We discuss the theoretical basis of this approach and its implementation within the CPML system.

1 Introduction

An important goal for web-page design is to provide viewer-oriented personalization of web-page content. Designers may strive to condition web-page content and appearance on the current preferences of the viewer, and probably on some underlying structure of the web-page content. For example, when the user is viewing an article about the consequences of a traffic accident in an on-line newspaper, the designer may wish to present a Volvo ad, as well.

Much of the content personalization literature focuses on learning user profiles. Although this technique is useful, it generally suffers from low availability, and tends to address only long-term user preferences. These schemes are thus applicable only to frequent viewers, that are, in addition, amiable to having information about their behavior managed by an external agent.

We propose a new model for representing web-page content. 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 taste is an important factor in how the web-page

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, we use well-founded tools for preferences elicitation and preference optimization grounded in qualitative decision theory. These tools help the designer structure her preferences over web-page content off-line, in an intuitive manner, and support fast algorithms for optimal configuration determination.

A web-page is typically composed of several components. The information content of each component can be either presented or not. In our model, the first step is for the web-page designer to express her preferences regarding the presentation of the web-page content. For example, the author may prefer some material to be presented if and only if some other material is not presented. This is done in an intuitive yet expressive manner using the CP-network (short for *ceteris paribus* networks) [Boutilier *et al.*, 1999] which are an intuitive, qualitative, graphical model of preferences that captures statements of conditional preferential independence. The description of these preferences, as captured by the CP-net, becomes a *static* part of the web document, and sets the parameters of its initial presentation. Then, for each particular session, the actual presentation changes *dynamically* based on the user's actual choices. These choices exhibit the user's content preferences. They are monitored and reasoned about during each session. No long-term learning of a user profile is required, although it can be supported.

Using this approach, we achieve content personalization through dynamic preference-based reconfiguration of the web-page. Whenever new user input is obtained (e.g., a click indicating his desire to view some item), the configuration algorithm attempts to determine the best presentation of all web-page components with respect to the web-page designer's preferences that *satisfies* the user's viewing choices. This process is based on an algorithm for constrained optimization in the context of a CP-network.

Determining only preferred content is generally insufficient – the chosen configuration should be presentable w.r.t. the layout constraints of the viewer's browser. Likewise, a web-page designer may wish to specify expectations of the exact appearance of the document, in addition to preferences about content. Declarative specification of the desired layout of a web-page are well known: cascading style sheets were introduced as a part of the HTML 4.0 standard [Lie and Bos,

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

1997], constraint style sheets were proposed in [Borning *et al.*, 2000].¹ We extend our content personalization approach to handle different layout constraints on web-page rendering, providing the viewer with a preferentially optimal feasible presentation of the web-page components.

We implement our approach in the framework of the *Ceteris Paribus* Markup Language (CPML) system, which consist of an authoring tool for the preference-based web-pages, and a corresponding viewing tool, which is implemented as a browser plugin.

The paper is organized as follows: Section 2 presents the framework of web-page preference-based configuration in the context of qualitative decision making. Section 3 discusses relevant preference representation issues, and describes the CP-network model. Section 4 describes the basic architecture and implementation of CPML. Section 5 illustrates the approach by example. Section 6 extends CPML by integrating layout constraints into the web-page optimization process.

2 Configuration and Qualitative DT

Any web-page can be considered as a set of components $C_1, \dots, C_n$. Each component is associated with its content. For example, the content of a component may be a block of text, an image, etc. In our work, each component may be either presented to the viewer or hidden, and these options for C_i are denoted by c_i, c'_i . However, the model can be expanded to handle more options for components' content presentation.

The web-page components define a configuration space $\mathcal{C} = \{c_1, c'_1\} \times \dots \times \{c_n, c'_n\}$. Each element σ in this space is a possible configuration of the web-page content. Our task will be to determine the preferentially optimal configurations, and to present one of them to the viewer. In terms of decision theory, the set of components of web-page is a set of features, the optional presentations of component's content are the values of the corresponding feature, and configurations are outcomes, over which a preference ranking can be defined.

First we define a preference order $\succeq$ over the configuration space: $\sigma_1 \succeq \sigma_2$ means that the decision maker views configuration σ_1 as equal or more preferred than σ_2 . This preference ranking is a total preorder, 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 web author. The author is the content expert, and she is likely to have considerable knowledge about appropriate content combinations. We would like the web-page to reflect her expertise as much as possible.

During the creation of the web-page, the designer describes her expectations regarding content presentation. Therefore, the preference order $\succeq$ represents the static subjective preferences of the web-page designer, not of its viewer. Thus, preference elicitation is performed on the web-page designer

off-line once for all subsequent accesses to the created web-page. The dynamic nature of the web-page stems from the interaction between the statically defined author preferences and the constantly changing content constraints imposed by recent viewer choices.

The choice of a representation model for the preferences is crucial in any preference driven configuration process. In particular, one can adopt either a quantitative, utility-theoretic, model, or a qualitative model. We believe that qualitative models can form a good basis for the automated product configuration in general, and for the web-page configuration in particular. For a comprehensive overview of the field of qualitative decision theory see [Doyle and Thomason, 1999]. The main advantages of qualitative decision theory tools, as opposed to traditional decision theory, are compactness, intuitiveness – which can considerably reduce the preference elicitation burden – and potentially reduced computational effort. Thus, in our domain, designers are likely to find (quantitative) utility assessment of content configurations unintuitive. Yet, the web-page designer is likely to be able to (qualitatively) compare and to rank alternative content presentations.

3 Preference Representation

Although a qualitative representation of preferences is typically simpler to represent and manipulate, the preference elicitation stage can still be quite complicated. To perform real-life preference-based configuration, we must represent user preferences in a compact, yet expressive manner. Even relatively small web-pages may consist of numerous components, making an explicit (exponential size) ranking table for all alternative content configurations impractical. Likewise, we wish to capture conditional preference dependencies between different components. For example, the designer may prefer to expose a short IJCAI call for papers (CFP) if the viewer explicitly examined the content of an article about a new book on AI, but not to expose the CFP otherwise.

An appropriate representation of preferences is insufficient on its own; we need to be able to reason about them efficiently. Reasoning tasks include finding preferentially optimal configurations, comparing two configurations. Likewise, the representation model should be intuitive in order to simplify the web-page designer's task.

Because of these requirements, in our work we decided to exploit the advantages of the CP-network model developed in [Boutilier *et al.*, 1999]. This 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 web-page designer to specify, for any specific component C_i of interest, content presentation of which other components $\Pi(C_i)$ can impact her preferences over the presentation options of C_i . For each content configuration π of $\Pi(C_i)$, the designer must specify her preference ordering over the presentation options of C_i given π . For example, suppose that the designer determines that $\Pi(C_i) = \{C_j, C_k\}$ and that c_i is preferred to c'_i given c_j and c'_k *all else being equal*. This means that given any two configurations that agree on all components other than C_i and in which C_j is

¹For a discussion on using constraints in user interfaces and interactive systems we refer to [Borning *et al.*, 2000].

IN and C_k is OUT, the configuration in which C_i is OUT is preferred to the configuration in which C_i is IN.

CP-networks bear a surface similarity to Bayesian networks: their representation is by a directed acyclic graph, with additional information - for each possible state of each node's predecessors. A node in the graph stands for a feature, which here represents a component C_i of the web-page. The immediate predecessors of C_i in the graph are associated with $\Pi(C_i)$. Formally, if $\bar{C}_i = \{C_1, \dots, C_n\} \setminus \{C_i, \Pi(C_i)\}$ then C_i and $\bar{C}_i$ 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, which describes the preferences about the values of the corresponding feature C_i given all possible combinations of $\Pi(C_i)$.

The acyclicity requirement is necessary, as introduction of cycles may cause an inconsistent preference structure. An effective algorithm for detecting such a problem does not yet exist, and we believe that the problem is NP-hard.

An example CP-network with the corresponding preference table is shown in Figure 1. We see that the designer specifies unconditional preference for presenting the content of component C_1 (denoted in figure by $c_1 \succ c'_1$). However, if C_1 content is presented and C_2 content is not, then the designer prefers not to expose the content of C_3 (denoted by $(c_1 \wedge c'_2) : c'_3 \succ c_3$).

In our domain, the interactive construction of a CP-network for web-page's content consists of two stages: for each component C_i of interest, asking the web-page designer to identify $\Pi(C_i)$, and to specify the C_i 's preference table. Note that each complete assignment for the nodes of the network represents a configuration of the web-page content.

Suppose that there are no constraints on web-page rendering. Then all content configurations are feasible. In this case, given a CP-network representation of the preferences, finding preferentially optimal configuration is straightforward: Traverse the nodes of the network according to a topological ordering and set the value of the processed node to its preferred value, given the (already fixed) values of its predecessors. The more complicated case of constrained configuration space $\mathcal{C}$ will be discussed later in this paper.

4 CPML System Description

This section presents a framework for preference-based web-page configuration, and shows how decision theoretic tools provide a basis for this application. Past work has dealt with general preference-based configuration (e.g., [Boutilier *et al.*, 1997; D'Ambrosio and Birmingham, 1995], but its adaptation to information personalization has not been explored. A prototype system (CPML) has been developed at our university, and here we describe some of its implementation issues.

The process of a partly unsupervised, preference-based web-page content configuration can be divided into 3 stages:

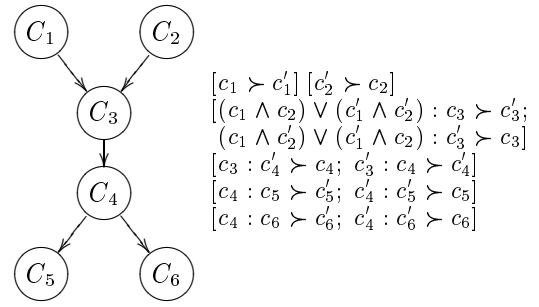


Figure 1: An example CP-network

1. Identify the components on whose content presentation the designer has a preference.
2. Apply interactive elicitation of the designer's preferences for the presentation of the selected components.
3. Reason about the specified preferences during viewer interaction with the web-page.

The CPML prototype system for this process consists of two modules - the *authoring tool*, and the *viewing tool*. The central part of the authoring tool is a specification of the CP-network for the created web-page (steps 1 and 2). Given such a specification, 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 web-page. Likewise, the authoring tool allows an optional specification of layout constraints on the web-page rendering. The nature of these constraints and their integration into the process of content optimization is described in details in Section 6. The outline of the system is presented in Figure 2.

First, the designer creates an HTML web-page. This task is standard for most web-page authoring tools. However, HTML document component contents and their representation are indivisible. Therefore, during step 1 (see below), the specified HTML components are automatically wrapped by javascripts [Rule and Rule, 1998]. This will allow presentation of components in different manners. Recall that this

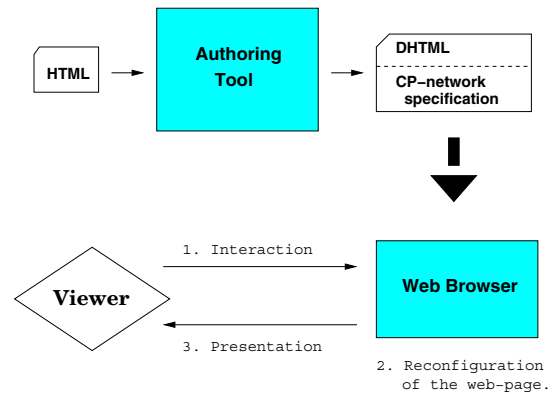


Figure 2: Framework of the CPML system

prototype system is restricted to only two options: presenting, or hiding, the component contents. The whole process of HTML expansion is completely transparent to the designer.

Next, step 2 in our approach is the definition of a CP-network over the specified components. For this purpose, an editor for CP-networks is integrated with the authoring tool. The result of preference-based web-page design, is a document specifying both what to present and how to present.

In turn, the viewing tool on the client side consists of a standard browser expanded by a CPML agent, implemented by a browser-compatible plugin. Accessing a preference-based web document in our system results in shipping the web-page to the browser and the embedded CP-network to the agent. Upon downloading the document, the agent sets all the components to their values in the optimal configuration, given no evidences on the viewer's current interests. From this point the agent acts in event-driven fashion, and waits for a viewer interaction with the browser. The user may either choose to expose or hide a component. The agent is responsible for reacting to viewer actions by reconfiguring the content presentation of the web-page. More specifically, the new content presentation should be a Pareto-optimal configuration of the web-page that contains (or omits) the item selected by the user for presentation (or for omission, respectively).

Determining the best configuration is done by the procedure presented in Figure 3. The simplicity of this algorithm

Procedure Reconfiguration

$\mathcal{G}$ - CP-network of the web page's components

$\mathcal{E}$ - Queue of the recent events (component, value)

Let e_i be the recent event: an interaction of the viewer with component C_i (setting on/off). Then

1. Switch the value of C_i and add e_i into $\mathcal{E}$. If $\mathcal{E}$ is bigger than some threshold - remove the oldest event from $\mathcal{E}$.
2. Project $\mathcal{E}$ on $\mathcal{G}$ (specify values of recently observed components w.r.t. $\mathcal{E}$).
3. Traverse $\mathcal{G}$ in a topological order and set each unspecified component to its most preferred value w.r.t. to the values of its predecessors in $\mathcal{G}$.

Figure 3: Preference-based Reconfiguration

stems from an important property that emerges from the semantics of CP-networks: parent preferences are more important than children preferences. This sanctions a fast top-down traversal process for generating optimal configurations.

As the CPML prototype is restricted to components with only two presentation options, in step 1 of Reconfiguration, we *switch* the value of the observed component, and then add it to the list $\mathcal{E}$ of the recent viewer's choices. The maximal size of $\mathcal{E}$ is specified by the designer of the web page and is passed as a part of the document. In step 2, we specify the values of the recently observed components as constraints on the required optimal configuration of the web-page. This is done in order to personalize the content to the viewer by reflecting her recent choices in content configuration. Finally, in step 3, given the constrained components, we determine the optimal configuration of the web-page. Due to the semantics

of the CP-network model, this process can be performed in time linear in the number of components.

5 Example

We illustrate the process through the following example. The designed web-page consists of seven components: four short articles, and three commercials. The articles are about the ongoing elections (Elections), a traffic accident (Traffic Accident), a new car airbag (New Airbag), and the results of the recent NBA games (NBA). The commercials are for New-York Times magazine (NY Times), Volvo cars (Volvo), and Nike shoes (Nike). After specification of the web-page content, the designer expresses her preferences about the content presentation:

1. By default, presenting the central article Elections (C_1) is preferred to hiding it (i.e. C_1 **on** is preferred to C_1 **off**. For the secondary article Traffic Accident, C_2 **off** is preferred to **on**.
2. Article New Airbag (C_3): C_3 **on** is preferred only if Traffic Accident is **on** and Elections is **off**.
3. Article NBA (C_4) **on** is preferred only if Traffic Accident is not presented.
4. Commercial NY Times (C_5) is preferred only if both Elections and Traffic Accident are presented.
5. Commercial Volvo (C_6) **on** is preferred if either New Airbag or Traffic Accident are presented.
6. The commercial Nike (C_7) **on** is preferred only if NBA is **on**.

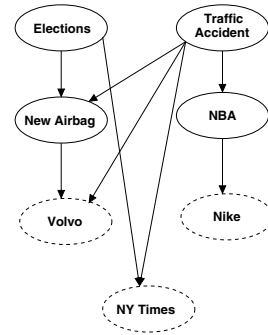


Figure 4: Example CP-network

The corresponding graph of the CP-network is presented on Figure 4 and the CP-tables are as follows:

$$\begin{aligned}
 C_1 &\Rightarrow c_1 \succ c'_1 \\
 C_2 &\Rightarrow c'_2 \succ c_2 \\
 C_3 &\Rightarrow (c'_1 \wedge c_2) \rightarrow c_3 \succ c'_3; (c_1 \vee c'_2) \rightarrow c'_3 \succ c_3 \\
 C_4 &\Rightarrow c_2 \rightarrow c'_4 \succ c_4; c'_2 \rightarrow c_4 \succ c'_4 \\
 C_5 &\Rightarrow (c_1 \wedge c_2) \rightarrow c_5 \succ c'_5; (c'_1 \vee c'_2) \rightarrow c'_5 \succ c_5 \\
 C_6 &\Rightarrow (c_2 \vee c_3) \rightarrow c_6 \succ c'_6; (c'_2 \wedge c'_3) \rightarrow c'_6 \succ c_6 \\
 C_7 &\Rightarrow c_4 \rightarrow c_7 \succ c'_7; c'_4 \rightarrow c'_7 \succ c_7
 \end{aligned}$$

Upon downloading this web document, the initial presentation of its content, depicted in Figure 5(a), is determined

by the Reconfiguration procedure. In the figure shaded nodes stand for presented components, and the rest for hidden components. Suppose that the viewer clicks on the link to the Traffic Accident article. Recall that such an interaction involves not just the presentation of the directly selected component, but the entire web-page presentation is reconsidered. The result of the reconfiguration is shown in Figure 5(b). Now, suppose that the viewer consequently clicks on the Elections component, trying to express that she is not interested in this topic². If the event queue was specified by the designer to contain only one, most-recent event, then previous interaction with the Traffic Accident component is removed from the queue, and the result of the subsequent reconfiguration is presented in Figure 5(c). Otherwise, reconfiguration is performed w.r.t. the last two interactions (Figure 5(d)).

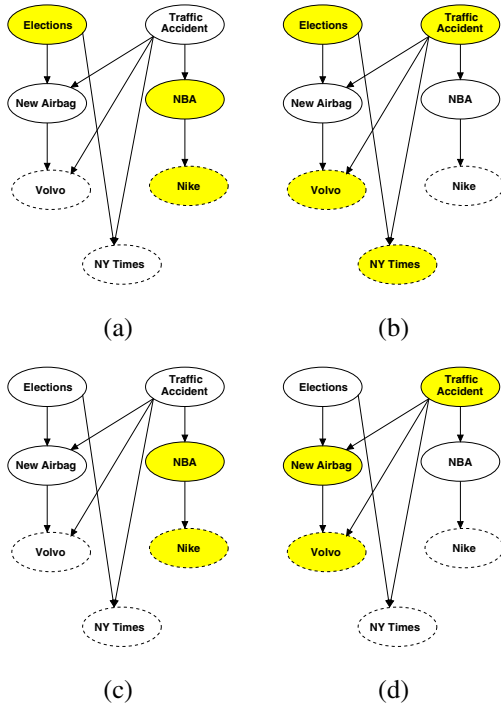


Figure 5: Reconfigurations of the content

6 Adding Layout Constraints

As mentioned above, the designer may specify expectations about layout of the presented components, as well as preferences about content presentation. The designer is allowed to specify layout constraints on web-page components, and we integrate these constraints into our preference-based optimization. We begin by discussing adopted layout constraints issues, followed by the modified reconfiguration process.

²Obviously, clicking on a component as expression of loss of interest is just one type of possible interaction - other kinds of interaction can be considered.

Using the CPML authoring tool, a designer can specify different constraints on the components of a web-page. Each component C_i is considered to be rendered in a rectangular area R_i , with neither its size nor its actual placement specified *a priori*. In turn, the designer is able to restrict the size and the form of each R_i , and to specify their relative placement and alignment. All these constraints are expressed as linear equalities and inequalities over the coordinates of the corners of the R_i s. The required solution for them should minimize an objective function that expresses the size of the rendering area³. This problem is an instance of linear programming, and thus can be solved in polynomial time.

Unfortunately, the designer may specify only a few layout constraints, thus we expect our problems to be underconstrained. In conjunction with the frequently used objective function that attempts to minimize the rendering area of the web-page, this will result in a spatial overlap of some of the components, an unacceptable situation in our domain. General overlap prevention for the unconstrained rectangular areas cannot be expressed as a linear program. We express the non-overlap constraint as a set of linear inequalities over integers as follows. For each pair of rectangles R_1 and R_2 of undetermined relative position, denote their spatial extents by $\{x_1^l, x_1^r, y_1^t, y_1^b\}$ and $\{x_2^l, x_2^r, y_2^t, y_2^b\}$ respectively. Here l, r, t and b stand for left, right, top and bottom respectively. The non-overlap of R_1 and R_2 is expressed by the disjunction:

$$x_1^l - x_2^r \geq 0 \vee x_2^l - x_1^r \geq 0 \vee y_1^b - y_2^t \geq 0 \vee y_2^b - y_1^t \geq 0$$

We rewrite this disjunction as the following set of inequalities, by introducing four new integer variables:

$$\begin{aligned} x_1^l - x_2^r &\geq -K_1 \cdot M & x_2^l - x_1^r &\geq -K_2 \cdot M \\ y_1^b - y_2^t &\geq -K_3 \cdot M & y_2^b - y_1^t &\geq -K_4 \cdot M \\ K_1 + K_2 + K_3 + K_4 &\leq 3 \end{aligned}$$

where M is a “sufficiently large” constant, and $K_i \in \{0, 1\}$ for $1 \leq i \leq 4$. These inequalities are added to other linear constraints specified by the web-page designer, and solved by a general purpose ILP solver. The additional constraints make the general layout optimization problem NP-hard. However, we do not expect that to cause intractability in practice, due to the usually restricted number of components in a web-page. During the preliminary evaluation of the CPML prototype, no significant response time degradation was observed.

As with CP-networks creation, our authoring tool provides a simple user interface that supports icon-based layout constraint specification. The web-page designer is not aware of the underlying linear expressions, which are generated automatically. These linear expressions automatically become a part of the created web document, together with the actual web-page and the CP-network’s description. Upon document download, these embedded constraints are passed to the CPML agent (together with the embedded CP-network), and become a part of the optimization process.

During user interaction, optimal layout w.r.t. the objective function is chosen for the configuration determined by the Reconfiguration procedure. The obvious question now is: what

³Other objective functions are also supported.

happens if the constraint system is unsatisfiable for a given, or even maximally possible, width and height of the viewer's browser? The first possibility is to use scroll bars. The second option is to try to compromise a little in the preferential optimality of the presented content. If we could determine a content presentation, which is both close to optimal w.r.t. the preferences of the designer and the viewer, and feasible w.r.t. the layout constraints, it may be a better solution.

One of the properties of the CP-network model is that given an outcome σ on can easily determine a set of outcomes $\{\sigma_1, \dots, \sigma_m\}$ such that, for $1 \leq i \leq m$, $\sigma \succ \sigma_i$ and there is no other outcome σ' such that $\sigma \succ \sigma' \succ \sigma_i$. In other words, $\{\sigma_1, \dots, \sigma_m\}$ is a set of *all* outcomes which are less preferred than σ but with a minimal loss of preference. The corresponding *worsening search* procedure is presented in [Boutilier *et al.*, 1999]. Informally, each outcome σ_i is reached by changing value of a single feature f in σ , such that $\sigma(f) \succ \sigma_i(f)$ given the same values of $\Pi(f)$ in σ and σ_i .

Procedure ConstrainedReconfiguration (e_i)

$\mathcal{G}$ - CP-network of the web page's components

$\mathcal{E}$ - Queue of the recent events (component, value)

$\mathcal{L}$ - Set of layout constraints

1. σ = Reconfiguration
2. If an $l = \text{ILPsolve}(\mathcal{L}, \sigma)$ exist, then present σ according to the layout specification l . Otherwise:
 - (a) Using restricted worsening search starting from σ , find the frontier of preferentially Pareto-optimal configurations $\mathcal{C}_o$ consistent with $\mathcal{E}$ that satisfy $\mathcal{L}$.
 - (b) If $\mathcal{C}_o = \emptyset$ then remove the oldest event from $\mathcal{E}$ and continue from 1. Otherwise, present a $\sigma' \in \mathcal{C}_o$.

Figure 6: Constrained reconfiguration

The extended version of the Reconfiguration procedure is shown in Figure 6. Note that the number of preferentially optimal, feasible configurations may be exponential in the number of features, thus in step 2a we explicitly restrict ourself in the worsening search. Consequently, if such a compromise in the designer's preferences does not result in a feasible configuration, we can allow ourself less attention to the preferences of the viewer. This is done in step 2b by explicitly ignoring some of the oldest viewer's choices. Naturally, all these parameters of the reconfiguration process can be specified by the web-page designer.

7 Summary and Future Work

A framework for preference-based configuration of the web page content was presented. Our approach is based on qualitative decision theory, and in particular on the CP-network graphical model for preference representation. The choice of qualitative, well-founded tools leads to an application that is both intuitive and fast. The configuration process is performed according to the preferences of the web author, as well as to the current interests of the viewer. Preferences of the web-page author are encoded as a CP-network. Given this CP-network and the recent actions of the viewer, the optimal configuration of the web page content is determined.

Likewise, geometric constraints on web-page layout are integrated within the optimization process, and different aspects of this integrated optimization were discussed. A prototype system was implemented, consisting of an authoring tool on the web-page author's side, and a decision making agent on the client side.

In future work, we plan to deal with a number of issues. First, preferred presentation of a web component may depend on the presentation of its neighbors, whose preferred presentation depend on its presentation, in turn. The resulting cyclic preference graphs are not well understood. The CP-network model presented in [Boutilier *et al.*, 1999] requires an acyclic graph, as the consistency of cyclic preference graphs is not guaranteed and depends on actual values in the preference tables. Thus, we plan to investigate the computational aspects of consistency verification in cyclic preference graphs.

Second, we wish to exploit our domain-specific knowledge in order to achieve an efficient ILP solver for our specific domain, in a manner similar to [Badros and Borning, 1998]. Likewise, we will consider translating some geometric constraints into actual constraints on the content presentation options of the web components.

Finally, specifying the behavior of intelligent user interfaces and autonomous multimedia objects is another potentially important application of qualitative preference models. Here as well, qualitative constraint-based optimization techniques seem useful.

References

- [Badros and Borning, 1998] Greg J. Badros and Alan Borning. The Cassowary Linear Arithmetic Constraint Solving Algorithm: Interface and Implementation. Technical Report 98-06-04, University of Washington, 1998.
- [Borning *et al.*, 2000] Alan Borning, Richard Lin, and Kim Marriott. Constraint-Based Document Layout for the Web. *ACM Multimedia Systems Journal*, 8(3):177–189, 2000.
- [Boutilier *et al.*, 1997] C. Boutilier, R. Brafman, C. Geib, and D. Poole. A Constraint-Based Approach to Preference Elicitation and Decision Making. In *AAAI Spring Symposium on Qualitative Decision Theory*, Stanford, 1997.
- [Boutilier *et al.*, 1999] C. Boutilier, R. Brafman, H. Hoos, and D. Poole. Reasoning with Conditional Ceteris Paribus Preference Statements. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, 1999.
- [D'Ambrosio and Birmingham, 1995] Joseph D'Ambrosio and William Birmingham. Preference-Directed Design. *Journal of Artificial Intelligence in Engineering Design, Analysis, and Manufacturing*, 9:219–230, 1995.
- [Doyle and Thomason, 1999] J. Doyle and R.H. Thomason. Background to Qualitative Decision Theory. *AI Magazine*, 20(2):55–68, 1999.
- [Lie and Bos, 1997] H.W. Lie and B. Bos. *Cascading Style Sheets*. Addison-Wesley, 1997.
- [Rule and Rule, 1998] Jeff Rule and Jeffrey S. Rule. *Dynamic Html: The Html Developer's Guide*. Addison-Wesley, November 1998.