Web Page Personalization to Improve e-Accessibility for Visually Impaired People

Yoann BONAVERO, Marianne HUCHARD and Michel MEYNARD
LIRMM, CNRS and University Montpellier II, Montpellier, France
bonavero or huchard or meynard @lirmm.fr

***Abstract*—Today's assistive technologies aim to improve the quality of information acquisition for visually impaired persons. Screen readers and screen magnifiers are the main visual assistive technologies that are currently proposed. However, due to economical reasons, these technologies do not consider specific needs of visually impaired people. In this paper, we propose an approach to make Web pages more accessible for users who have specific needs. User wishes, like text color, font size, link color or even more complex wishes including wishes about brightness or contrast are encoded as user preferences. We also encode designer graphical choices as designer preferences. From these preferences, a new Web page design is computed using a resolution algorithm. We compare two distinct approaches to make this computation: resolution algorithms from preference theory domain and an evolutionary algorithm. The comparison evaluates running time on automatically generated data and on the sign in Facebook page.**

Keywords- Web page; Personalization; visually impaired; evolutionary algorithm.

I. INTRODUCTION

About 285 million people are estimated to be visually impaired worldwide (39 million are blind and 246 have low vision) [1]. These figures are constantly growing mainly because of increased life longevity due to medical advances. In France, people with disabilities, especially visually impaired people, use internet more frequently and have more computer devices or interfaces than the average French sighted population [2].

Information and communication technologies are increasingly used by everyone in everyday life. Unfortunately, it can be a double-edged issue for people with visual impairment. On the one hand, these technologies offer many solutions for current life activities like online purchase or administrative services. They give access to information that was previously inaccessible. On the other hand, many issues remain, due to the design, development and technologies used to make the website content. As a consequence, these technologies which are able to compensate user disabilities can also be a new source of exclusion and discrimination.

Today's technological environment offers numerous online services and multiple data sharing capabilities. This data is distributed in many places and under different formats. Data can be a Web page, text based document, images, video, sound, etc. Concerning websites, data is displayed on pages according to a visual style defined by the author. The different choices made by the designer create the graphical context of the page which reflects the brand or the organization. This graphical design is intended to influence the reader to recognize, assimilate, guide, memorize a page and associate it with the related brand or organization. It also determines the page understanding and user's navigation, and it is intended to help the user in these tasks.

The W3C consortium publishes sets of technical specifications to make accessible websites. The conformity of websites to these specifications assumes that they can be used by assistive technologies. Tools and guidelines are provided to developers and end users; Web Content Accessibility Guideline (WCAG 2.0 [3]), and other guidelines including User Agent Accessibility Guideline (UAAG [4]), Authoring tool Accessibility Guideline (ATAG [5]) and the Web Accessibility Initiative - Accessible Rich Internet Application (WAI - ARIA [6]). Organizations, like BrailleNet, have been created to operationalize the different standards of the W3C guidelines including "AccessiWeb". Unfortunately e-accessibility is not a main concern of website designers and developers. It is often considered as a waste of time or an additional development cost, giving unsightly results which are only targeting a small part of the population.

Assistive technologies have existed for several years and are widely used by disabled people. Screen readers allow the user to get information in another communication way: vocal synthesis or braille display are used to vocalize information or display it in braille. Visually impaired people with low vision, use their partial sight as the principle means to access information. Screen magnifiers are applications that improve visual comfort and increase information acquisition. With these tools, it is possible to use zoom and color filters to compensate visual issues. These tools are useful but not sufficient to ensure e-accessibility because they have a general purpose, and they are not adapted to specific needs. This is mainly due to an important maintenance cost to maintain compatibility with some applications like browsers, and to after-sales technical issues.

In this paper, we address the problems of adapting Web page design to the specific needs of a visually impaired person. Our approach proposes replacing the current pixel-level treatment process (in magnification filters) by an adaptation process based on the knowledge of the HTML element types and their properties. The adaptation is performed from a set of user wishes also called preferences. These preferences are the basis for a dynamic adaptation. Firstly, the users specify their specific needs in terms of wishes. Then, some Web page elements

 ISBN: 978-1-61208-333-9

are adapted by solving the preference set. In Section II, we explain how existing visual assistive technologies work, and we highlight their main drawbacks. Section III presents our approach in detail and proposes a user wishes representation. We present in Section IV the user's preference theory as a basis of this work. We develop in Section V a new approach allowing us to deal with real simple pages. We conclude in Section VI.

II. Existing work and problem statement

In this section, we explore several hardware and software solutions developed to improve website accessibility for visually impaired people with low vision. We distinguish between two main categories of approaches and tools. Some of them are used as development tools whilst others are used by end-users.

The World Wide Web Consortium (W3C) is the origin of HTML and CSS standardization. It also works on accessibility via several initiatives including the Web Accessibility Initiative - Accessible Rich Internet Application (WAI-ARIA [6]). These standards have evolved through many versions to address the emergence of new technologies. They have two main objectives. The first is to ensure that resources can be parsed and used by external assistive technologies. The second is to provide a minimal access to contents for people who don't use assistive technologies. These standards do not target a specific programming language (HTML, JavaScript, CSS, etc.). In addition to the standards, guidelines like WCAG [3], UAAG [4] or ATAG [5] framework and tools are published to ease the use of the standards. We can mention "AccessiWeb", developed by the BrailleNet organization which provides a simple operational interpretation of standards. The rules and standards are classified according to their importance in making websites accessible. Three increasing accessibility levels have been defined (A, AA and AAA). The first level (A) gives basic mandatory advice to ensure information accessibility. The second level (AA) notifies important recommendations to be respected to avoid difficulties in accessing information. The third and latest level (AAA) is about additional and optional ways to improve information access quality. When Web designers and developers include accessibility dimension in their websites, they mainly try to reach the intermediary AA level. Only a few websites which are very specialized require the highest level AAA.

Many existing tools allow developers to improve accessibility of their websites. These tools analyze the HTML source code and automatically rewrite it, or assist the developer to correct it in accordance with the standards [7]. These tools can be split into two categories: evaluation tools and transformation tools. The main drawback of these tools is that they do not permit adaptation for all user's needs. Some user needs can contradict each other. Conflicts can appear due to dependencies between needs. Consequently, automated evaluation and transformation tools can only assist developers to reach a general scope accessibility but are limited to implementation of the minimum recommended by standards.

Some kinds of applications which are improving accessibility tools are available to get information from websites and report it to the user through another communication protocol. For example, for users with low vision, it is possible to retrieve information by transforming visual output with magnification applications or accessibility browser options and extensions. Magnification programs allow the user to zoom on windows. Some of them propose font smoothing to avoid blurred characters, and mouse pointer modification to improve tracking movements. As a last example, magnification programs can apply filters on the window. Filters include "grey scale", "one color scale", "black and white", or "color inversion". The browser options and extensions enable us to manipulate style sheets. It is possible to completely remove style sheets or to define a unique style sheet that will be applied to all Web pages. To facilitate modifications, many browsers provide a graphical interface to help configuration of such properties as background color, text color and size or link color.

With the two solutions defined previously (screen magnifier and style sheet redefinition), we can theoretically adapt almost all pages to be suitable for a large part of the impaired population. However, in practice it is more complex. These pages could be even better modified for the population, and there is a requirement for additional tools to meet more needs of visually impaired people. For various reasons, the existing solutions are not suitable for everyone. For example, magnification tools perform treatments on global images. Hereafter, we develop a small practical example.

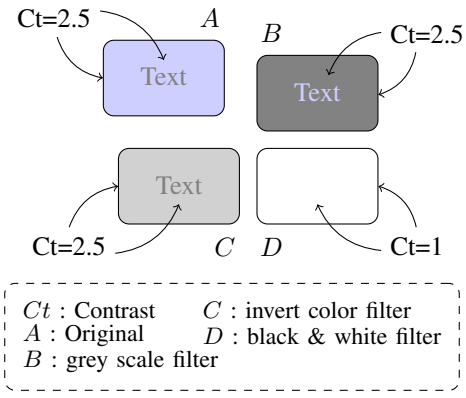

Figure 1. Filter application

Figure 1 illustrates in A an original text and its background. In B, we apply color inversion to A; C is obtained by applying grey scale filter to A; D is obtained by applying black and white filter to A. The contrast between two graphical elements is a value between 1 and 21, 1 is the null contrast and 21 the maximum contrast. The problem is that if there is low initial contrast between text and background, a filter cannot increase it to improve readability, except the black and white filter. However, in the "black and white" filter case, if both text and background are too light or too dark, both will become equal to white or dark. Thus the contrast is reduced to 1. This is due to the conversion threshold of the application when all

 ISBN: 978-1-61208-333-9

elements are both below or both higher than this threshold. To avoid this, the filter software should have an adjustable threshold but this is not often the case in practice.

The most currently used browsers have modules, extensions or simply options to transform Web pages. This is based on style sheet rewriting or disabling. Disabling the style sheet only keeps the content, and a lot of information about the context of the page, the brand image, etc. is lost. An inconvenience of this solution is that if developers don't respect a proper separation between source code and style sheet, several problems of overlap, missing, or unreadable information can occur. Style sheet rewriting is another alternative where the page design is changed. In this alternative, the user can define some properties like text color, background color, link color. More advanced users can define their own style sheet and use it for all the Websites they browse. This solution allows the user to keep the page layout and to change properties such as colors, but the context is still lost when important modifications have been done in order to compensate the user disabilities. Moreover, in this solution, the user often has to set many properties, including text color, link color, visited link color, hovered link color, title level 1 color, title level 2 color, etc. As a result, he has to manipulate a set of technical terms and a lot of options, and this task often is cumbersome and time consuming. Furthermore, these property changes have to be done through interfaces that may be difficult to manipulate by an impaired person.

Beyond these modules or extensions, some more evolved proposals try to enhance them with the widest physical characteristics configuration support [8]. In these applications, real time transformations are applied on Web pages in order to address these needs. These end-user side applications allow the user to configure text properties such as size, letter spacing, or line spacing, colors of the text, background and links. They also allow the user to configure the image display (show or hide) and table display. Once the modifications are applied, almost all information about website colors may disappear. Then, the chosen site ambiance may be lost. In our approach, our objective is to adapt a page in accordance with its original appearance, with its structure and with the user's preferences.

Even if some navigation problems can be avoided when webmasters respect all standards, statistics tell us that less than 10% of public websites would be fully accessible [9]. Other problems only depend on user vision. For example, if a user needs to have a dark text on a light background and if the page already contains both situations which are one menu with light text on dark background as well as a dark text on a light background, it is impossible with classical filters to get a relevant adaptation. If we use inversion color filter to make the menu accessible, the page content becomes inaccessible. Moreover, this kind of manipulation can generate dazzle from other parts of the website.

This last example shows us that it is not relevant to deal with a global image. To adapt the previous page we have to split it into two parts, the menu and the content, thus considering the type of the manipulated elements. We propose in the next section a new approach to separately process page elements according to their semantics.

Two research works close to ours deserve to be mentioned. In [10], authors propose to configure Web page display according to user actions and behavior. In our case, we want to propose an accessible Web page using user's needs considering the initial page. In [11], they propose to personalize Web display (shopping gallery) to a specific user or user group. The analysis of user usage on existing websites allows user pages to be shown with different structure and navigation. In our case, we want to apply adaptation in the client-side on original pages which initially are identical for all users. Our approach is led by all these observations and aims at proposing a personal adaptation following the wishes of users. The main principles of this approach are developed in the next section.

III. Our approach

We aim to develop a general approach which is independent from specific pages and which is able to respond to any specific user's wish. This new approach attempts to solve the principles problems shown in the previous section.

To solve the global treatment problems described in section II, our approach considers the following four components: Objects and properties (HTML elements and style) of the page written in the HTML and CSS files; Variation points (for example the color of a specific element); User's wishes; Algorithms for finding an adaptation from initial Web page according to the user's wishes.

Web pages are composed of HTML elements. These elements are organized within a tree structure. Each element has physical properties that determine its appearance, including its size or its color and more abstract properties that define the element type (menu, content, image, etc.). From HTML 5 version, there are tags to describe semantic types including navigation blocks (menu), articles, sections, ... Thus, if the page is implemented in HTML 5, we have adequate high-level information, but where the page is written in an older HTML version, it is necessary to detect some important parts of the page particularly the menu and the main content section.

We define *a set of objects* $\{O_1, O_2, .., O_n\}$ that represent all HTML elements that are important in the page modification process. Elements that will not be updated or that will not be used in computing are excluded from the object set.

The *variation points* are a set of variables $\{V_1, V_2, .., V_m\}$ induced by properties which are either basic properties written in the HTML or CSS files, or computed from these basic properties. For example, height, width or color are basic properties found in the HTML or CSS files, while area is a computed property derived from height and width.

To be able to change the value of the red color component of an object, a variable is added to the variation points. This variable domain is the value set of the red color component of the object. A pair composed of an object and a feature is associated with one and only one variable.

On these variables, a user can make choices that are called *preferences* or *wishes* $\{C_1, C_2, .., C_k\}$. For instance a user

 ISBN: 978-1-61208-333-9

can say: “I prefer light background to dark background for the main body page”. This preference only concerns the main body background and ignores the background of other page elements. With this page element segmentation we can define different preferences for each object. All choices made on one or more variables constitute user’s preferences. The main difference to existing work using user’s preferences is that our preferences are not direct values for features like in [8], but constraints to determine such values.

We have considered different levels in the description of preferences. The basic preferences are represented as $V_i\ op\ x_i\ >_p\ V_j\ op\ y_j$, where V_i and V_j are two variables (with possibly $V_i = V_j$), $>_p$ the preference symbol ($A >_p B$ means A preferred to B), op is a boolean operator like $=$ and x_i (resp. x_j) is in the domain of V_i (resp. V_j). To represent the user’s wish “I prefer black text to blue text”, we use the variable c_T to represent the color of a text object T. The domain of c_T is $\{white, red, blue, black\}$. The user’s wish is expressed as: $c_T = black >_p c_T = blue$.

This representation can be improved by adding conditional preferences to be able to represent: “I prefer bold font to normal font if the font color is yellow”. Here, we introduce a new variable w_T for representing the weight of the text, and a new operator ’:’ to represent the condition. This wish can be represented as $c_T = yellow : w_T = bold >_p w_T = normal$. This representation is considered in Section IV. In this paper we do not explore how to obtain the user’s wishes. It is another research element that will be explored in future work. For example, we may be able to ascertain a user’s wishes using a preference learning algorithm.

We also explored in Section V another extended representation in which we use any complex function on variables and their domain values. This allows us to express wishes like “I would like to have a text size higher or equal to 14pt” or “I would like to have a contrast between text and direct background higher or equal to 50%”. In this last example, the contrast is a binary function. It represents a distance between the colors of the two objects: the text T and the body B. We define two variables, c_T that represents the text object color and c_B that represents the text body color. We define a contrast function $contrast(x, y)$ that returns the contrast between x and y. The result of this function is compared to a user specified domain value, giving a complex wish, which is an evaluable expression: $contrast(c_T, c_B) \geq l$ where l is the required level.

We evaluated the size of the input data of our approach on a few sample websites. Depending on the Web page, the number of variables representing HTML elements associated with a basic feature can vary from around 10 to hundreds for the biggest websites. For example, among most visited websites, the “Facebook” registration page will be represented in our system by about 40 variables to implement the preference: “contrast between text and direct background higher or equal to x”. For this preference we have to extract all text elements and their direct backgrounds, making contrast variables. For the Google search page (not the result page) we obtain around 17 variables, while for “BBC News” home page we obtain around 200 variables. These values are rounded because large parts of websites are dynamic and have different elements each time a page is loaded (these computations have been made on november, 14, 2013).

Domains of variables may have several dimensions. For example, the text size variable domain generally has about 10 or 20 values. By contrast, the color variable domain can reach 2^{24} values in a true context. The number of variables depends on the type and the complexity of the given user’s preferences.

In the next section, we introduce preference theory as a theoretical framework for solving our problem. This theory also provides a comparative basis for another solving algorithm.

IV. Preference theory

The first approach that we considered to solve the adaptation problem is the preference theory approach [12]. In this section, we explain how user’s preference theory can solve Web page adaptation problems and we outline the main resolution process. We also explain why a direct implementation of this approach does not scale in our context.

Many representations [13], [14], [15] allow formal description of user’s wishes for solving different problems. Most of the approaches rely on basic and conditional preferences as defined in previous Section III. In our context, general preferences like "something preferred to something else" may have to be duplicated. For instance, for a text size variable V and its domain $\{6, 8, .., 14\}$, if the value 14 is preferred to the others we have to create the following preferences: $V = 14 > V = 6$, $V = 14 > V = 8$, $V = 14 > V = 10$, $V = 14 > V = 12$. The approach works on two data sets. The first set groups all variables, while the second set gathers the user’s preferences. Two ways of solving and finding solutions exist: an explicit resolution or an implicit resolution.

In explicit resolution, the first step consists of creating all possible adaptation solutions. In our context, this consists of creating all combinations of valued Web page properties. A solution is a set of valued variables, where each defined variable has a value. For example, if we only have the two variables c_T for text color and c_B for background color, a solution may be $\{c_T = blue, c_B = white\}$. The next step consists of producing for each preference, a pair of sets where the first set contains solutions satisfying the preference and the second contains solutions that satisfy the complementary preference. Solutions that do not satisfy the preference or its complementary are not included in the sets. For pages with over 10 elements of 10 or more values in domain, it is impossible to process this first generation step in a short time. By short time we mean an additional time to the page loading time that can be accepted by the user. If the user has to wait too long for each page to load, the application based on this approach will rarely be adopted. We implemented this solution to evaluate its feasibility.

The implicit resolution is slightly different. All combinations of variable sets and values don’t need to be generated.

 ISBN: 978-1-61208-333-9

Preferences are given in intent (using a description in a formula in propositional logic) and the resolution uses these intents as input parameters. Ultimately, we obtain a partially ordered set of expressions which describe the solutions. An example of such expression, for representing "a blue text and a minimal contrast of $x\%$ between text and background color" is $c_T = blue \wedge contrast(c_T, c_B) \geq x$. The initial resolution process is faster than the previous one. However, the difficulty has just been delayed. Now, a partially ordered set of expressions is produced, and we have to find in the whole set of solutions if a solution exists and if this is the case, which solutions correspond to these expressions. If top solutions contain a computed part like $contrast(c_T, c_B) \geq x$, then we have to calculate the contrast for each solution to determine if it has the required level.

The preference theory approach gives the best solutions in adapting Web pages but we show in the following section that, with a direct implementation of the approach, the running time exceeds what a user may find acceptable. Nevertheless, preference theory provides a theoretical framework and an initial way to achieve adaptation solutions. In the next section we present a second approach that allows us to deal with practical cases.

V. An evolutionary algorithm based approach

In order to face the scalability issues, we considered defining our problem as an optimization problem and to adopt a meta-heuristic approach. An evolutionary algorithm has been chosen to cross the whole solution set in order to find a "good" solution, using a set of basic modifications on a solution. These basic modifications must allow us to potentially reach any solution of the whole solution set, but only part of the solutions will be explored in order to reduce the computational complexity. Basic modifications are implemented by operators including selection, crossover or mutation. The chosen algorithm processes a local search trying to reach a best local solution.

Among evolutionary algorithms, we chose the Non dominated Sorting Genetic Algorithm II (NSGA-II [16]) which is a multi-objective genetic algorithm (MOGA). This feature allows us to separate basic objective functions into different main objectives and to obtain multiple Pareto-optimal solutions. NSGA-II is the evolution of NSGA [17], and it mainly reduces the complexity of the previous version. It has a computational complexity of MN^2 instead of MN^3 where M is number of objectives and N is the population size. A population is a set of solutions evolving through operators along algorithm execution.

NSGA-II is an iterative process. At the first iteration we create and randomly initialize a population P_0 that contains N solutions. Duplicated solutions may exist. Each individual is rated by objectives functions. We sort the population based on non-domination to obtain several solution groups called fronts. Rank one is assigned to the first front (solutions not dominated by any other solution), rank two to the second front (solutions dominated only by solutions of the first front) and so on. To each solution is assigned a distance value called crowding distance [18], which maintains diversity between the solutions. This distance is used by the selection operator to select solutions that may evolve thanks to crossover operator and mutation operator. The new population obtained after application of the operators is called offspring O. We compute the objective functions on each individual of O. The union operator combines the previous population P_0 and the offspring O to get a doubled size intermediate population. This intermediate population is reduced by keeping the N best solutions in terms of non-domination to give the next population P_1. This is repeated until the termination criteria is achieved.

In our context, we represent a solution as a list of valued variables. For example $\{V_1 = x, V_2 = y, V_3 = z \dots\}$ where V_1, V_2 and V_3 are three variables. x, y and z are respectively in the domains of V_1, V_2 and V_3. The crossover operator builds a new solution by composing two parent solutions. The new solution is composed of odd variables of the first parent solution and even variables of the second parent solution. For example, if the first parent is $\{V_{i_1}, V_{i_2}, V_{i_3}, V_{i_4}, V_{i_5} \dots\}$ and the second parent $\{V_{j_1}, V_{j_2}, V_{j_3}, V_{j_4}, V_{j_5} \dots\}$ the crossover operator builds $\{V_{j_1}, V_{i_2}, V_{j_3}, V_{i_4}, V_{j_5} \dots\}$. The mutation operator randomly chooses which variables of a solution will be mutated and assigns to these variables a value chosen randomly in their respective domains. The selection operator is based on crowding distance to select the solutions leading to the next population.

As said previously, such an algorithm does not compute the whole solution set. Rather, selection, crossover and mutation operators enable us to compute solutions in the neighborhood of the current considered solutions, trying to reach solutions that improve the objectives. In this approach the complexity induced by the huge number of adaptation solutions can be highly reduced. We implemented this NSGA II version in C++ in order to evaluate its scalability on our problem. Table I shows and compares on several datasets (A, B and $Facebook$) the running time of our implementations of the algorithm of preference solving and of the evolutionary algorithm NSGA-II. For the NSGA-II implementation running time, these results are the average of 400 executions. The results of Table I have been obtained on a common laptop with quad core (2.6Ghz) microprocessor assisted by 8GB of DDR3 RAM.

Table I
Running time of preference theory algorithms and NSGA-II

Data \ Algos	Pref. algos	NSGA-II
A	6 mn	< 1 ms
B	> 24h	500ms
$Facebook$	> 24h	500ms

The two first datasets A and B have been automatically generated in order to represent data that may be found in very simple Web pages. A dataset is composed of only 6 variables with 6 values in each domain. Some simple preferences were given between two values of the same or of different

variables. In this case, preference theory algorithms (explicit approach) have to generate the 6^6 solutions, which gives a running time of 6 minutes. Meanwhile, NSGA-II algorithm can find a "good" solution in less than one millisecond. Each objective function returns for a solution a quality value corresponding to one criterion. A good solution is a solution that satisfies all the user's wishes but is not especially the best one regarding objectives. Best solutions have the optimal value for all criteria. The second dataset (B) is also an automatically generated dataset with 8 color type variables and 32768 values in their domains. Here, with the first approach (preference theory), there is a combinatorial explosion (32768^8 solutions to be parsed) and we cannot obtain the results in less than one day, while only about 500ms are necessary for NSGA-II algorithm to get a good solution.

The last dataset is derived from the real Facebook home page which is accessed when a user is not authenticated (registry page). We consider the wishes "contrast between text and direct background is higher or equal to x%" with $x = 40\%$. The variables correspond to the text elements and their direct background. We obtain 36 variables and 24 binary relations representing wishes, and we consider 2^{15} domain values. The average running time is very close to the B case running time. The Facebook case contains strong dependencies between variables, for example, many texts have the same background and, as a consequence, they have the same associated background color variable. In this configuration, we can tune different parameters (population size, crossover probability) to reach a time as short as in the B case.

The results obtained using a direct implementation of the preference theory approach (explicit resolution) are disappointing, and we expect that implementing the implicit resolution would produce similar results, because the implicit resolution just moved the difficulty (generating all solutions) to the end of the resolution process. As the best solutions are described by an expression, we have to search these concrete solutions (if any exists) in the whole solution set, leading to a huge computational time. Fortunately, the results with NSGA-II are very encouraging. They show the feasibility of the approach on simulated data and on specific user preferences, corresponding to a common case of visual disability.

VI. Conclusion and future work

Existing assistive technologies help visually impaired people in accessing information. However, they do not provide fully relevant solutions for all kinds of visual disabilities and they can radically change the website appearance. To address the problem of adapting Web pages to specific needs, we investigated the use of the theoretical framework of preference theory. Nevertheless, while implementing the explicit resolution approach, we faced a scalability issue. It is worth noting that future research on preference theory may solve this problem. Then we explored the use of the evolutionary algorithm NSGA-II as an alternative approach. Even if they still require confirming on more datasets, the initial results are encouraging.

As future work we plan to investigate other resolution approaches. We plan to model our user's wishes as a Constraint Satisfaction Problem (CSP modelling) and to evaluate the performance of the existing constraint solvers in our context. Since our target is helping people with visual deficiency, we will test the approach on many frequently accessed Web pages and we will study how to facilitate the expression of preferences by the end users. A test with visually impaired persons will be organized.

Acknowledgment

The authors would like to thank Berger-Levrault which supported this work with a grant and Martine Hornby for her assistance in english revision.

References

[1] "Visual impairment and blindness, fact sheet n°282," World Health Org., http://www.who.int/mediacentre/factsheets/fs282/en, oct., 2013.

[2] G. Montagné, "Visually impaired and blind persons insertion in today's world (l'inclusion des personnes aveugles et malvoyantes dans le monde d'aujourd'hui), in french," *Rapport pour le Ministère du travail, des relations sociales et de la solidarité*, 2007, http://lesrapports.ladocumentationfrancaise.fr/BRP/084000321/0000.pdf.

[3] *Web Content Accessibility Guidelines*, World Wide Web Consortium, http://www.w3.org/TR/WCAG20/.

[4] *User Agent Accessibility Guidelines*, World Wide Web Consortium, http://www.w3.org/TR/UAAG20/.

[5] *Authoring tools Accessibility Guidelines*, World Wide Web Consortium, http://www.w3.org/TR/ATAG20/.

[6] *Web Accessibility Initiative - Accessible Rich Internet Applications*, World Wide Web Consortium, http://www.w3.org/WAI/intro/aria.

[7] M. Y. Ivory, J. Mankoff, and A. Le, "Using Automated Tools to Improve Web Site Usage by Users with Diverse Abilities," *Information Technology and Society*, vol. 3, no. 1, pp. 195–236, 2003.

[8] J. T. Richards and V. L. Hanson, "Web Accessibility: A Broader View," in *In WWW '04: Proceedings of the 13th international conference on World Wide Web*. ACM Press, 2004, pp. 72–79.

[9] K. Cullen, L. Kubitschke, T. Boussios, C. Dolphion, and I. Meyer, "Web accessibility in european countries: level of compliance with latest international accessibility specifications, notably wcag 2.0, and approaches or plans to implement those specifications (2009)," 2009.

[10] C. Domshlak, R. I. Brafman, and S. E. Shimony, "Preference-based Configuration of Web Page Content," in *Proc. of Int. Joint Conf. on Artificial Intelligence*, 2001, pp. 1451–1456.

[11] B. Mobasher, R. Cooley, and J. Srivastava, "Automatic personalization based on Web usage mining," *Com. ACM*, vol. 43, no. 8, pp. 142–151, 2000.

[12] S. Kaci, *Working with Preferences: Less Is More*, ser. Cognitive Technologies. Springer, 2011, iSBN:978-3-642-17279-3.

[13] C. Boutilier, F. Bacchus, and R. I. Brafman, "UCP-Networks: A Directed Graphical Representation of Conditional Utilities," *CoRR*, vol. abs/1301.2259, 2013.

[14] C. Boutilier, R. I. Brafman, H. H. Hoos, and D. Poole, "Reasoning with conditional ceteris paribus preference statements," in *Proc. of the Fif. conf. on Uncertainty in artificial intelligence*, ser. UAI'99, 1999, pp. 71–80. [Online]. Available: http://dl.acm.org/citation.cfm?id=2073796.2073805

[15] P. Haddawy and S. Hanks, "Representations for decision-theoretic planning: Utility functions for deadline goals." in *KR 1992*. Morgan Kaufmann, 1992, pp. 71–82. [Online]. Available: http://dblp.uni-trier.de/db/conf/kr/kr92.html#HaddawyH92

[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II," 2000.

[17] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary computation*, vol. 2, no. 3, pp. 221–248, 1994.

[18] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary computation*, vol. 10, no. 3, pp. 263–282, 2002.

 ISBN: 978-1-61208-333-9