Considérer les Besoins Propres à Chaque Individu pour Améliorer l'Accessibilité des Pages Web

Yoann Bonavero, Michel Meynard, Marianne Huchard

Considérer les Besoins Propres à Chaque Individu pour Améliorer l'Accessibilité des Pages Web

Yoann Bonavero[1], Michel Meynard, and Marianne Huchard

LIRMM, CNRS et Université de Montpellier
Yoann Bonavero <yoann.bonavero@lirmm.fr>

Résumé

Alors que les outils de l'information et de la communication occupent aujourd'hui une part très importante de notre vie quotidienne, l'accessibilité et l'utilisabilité de la multitude de services proposés ne sont bien souvent pas au rendez-vous. Ceci est d'autant plus vrai lorsque l'on a une pathologie lourde ou rare. Il est évident que des pages ou services Web ne peuvent d'eux-même pas être adaptés pour tout le monde et les outils tiers d'assistance sont souvent très généralistes. Nos travaux s'intéressent à une approche nouvelle permettant de prendre en compte le design initial ainsi que les attentes personnelles des utilisateurs afin d'adapter au cas par cas le visuel des pages Web au fil de la navigation. Cette approche s'appuie sur différentes tâches de découpage et de reconnaissance des structures dans les pages Web, d'identification des éléments posant des difficultés visuelles, puis sur une tâche de recherche d'une adaptation pertinente qui est enfin appliquée.

1 Introduction

La place grandissante qu'occupe aujourd'hui Internet en fait une source d'informations et de services considérable et incontournable. Les sites Web sont en constante évolution et les avancées et libertés offertes par les nouvelles technologies augmentent de plus en plus leur complexité. En appui à cette multiplicité de services et d'informations, la couverture Internet qui s'invite aujourd'hui presque partout creuse encore et encore les différences entre ceux qui peuvent accéder aux nouvelles technologies et les utiliser et les autres.

On considère les avancées médicales comme un progrès, un pas en avant, cependant cela dépend aussi de l'angle par lequel on les regarde. En même temps que les avancées médicales permettent d'augmenter l'espérance de vie moyenne, le nombre de personnes souffrant de déficience visuelle augmente lui aussi. Le risque d'avoir une pathologie visuelle avec l'âge est en effet important (DMLA, cataracte, ...). La problématique d'accès à l'information, à la culture, aux loisirs, aux services est par conséquent un enjeu majeur dans l'inclusion des personnes déficientes visuelles qui constituent une part croissante de la population.

Pour répondre aux besoins concrets, de nombreux standards et recommandations d'accessibilité ont été définis [2, 18, 19, 20]. Ils ont pour but d'assurer à la fois une accessibilité minimale à l'information mais également de permettre aux applications tierces d'assistance de fonctionner correctement. Ces outils, extérieurs aux sites Web et services, permettent d'apporter des modifications sur la façon dont les informations sont affichées à l'écran ou même sur la modalité d'accès à l'information. Les technologies tierces d'assistance, pour les personnes déficientes visuelles, peuvent être divisées en deux groupes distincts : celui des outils utilisés par les personnes qui possèdent encore un reste visuel utilisable pour accéder à de l'information visuellement et celui des outils utilisés par les personnes non-voyantes. Nous faisons donc ici la distinction entre les personnes non-voyantes qui utilisent par exemple des lecteurs d'écran et les personnes malvoyantes qui utilisent, par exemple, des agrandisseurs d'écran.

2 L'approche globale

Dans ses grandes lignes, notre approche consiste, lorsqu'une page Web est ouverte, à regarder comment la page est constituée (architecture, couleurs, dimensions, etc.) et à modifier des propriétés de certaines parties de la page pour qu'elle devienne accessible pour une personne donnée.

Il s'agit en effet de déterminer les zones des pages qui posent problème non pas pour une catégorie de personnes mais plutôt pour une personne individuelle afin d'adapter au mieux la page. Pour cela nous devrons également connaître les besoins de la personne. Le processus complet d'adaptation depuis une page Web par rapport aux besoins d'un utilisateur (cf. figure 1 peut se découper selon plusieurs grandes parties : l'analyse de la page Web et la sélection des éléments nécessaires à l'adaptation (cf. Section 3 et Section 4), la recherche d'une adaptation (cf. Section 5) et l'application sur la page Web de l'adaptation trouvée (cf. Section 6).

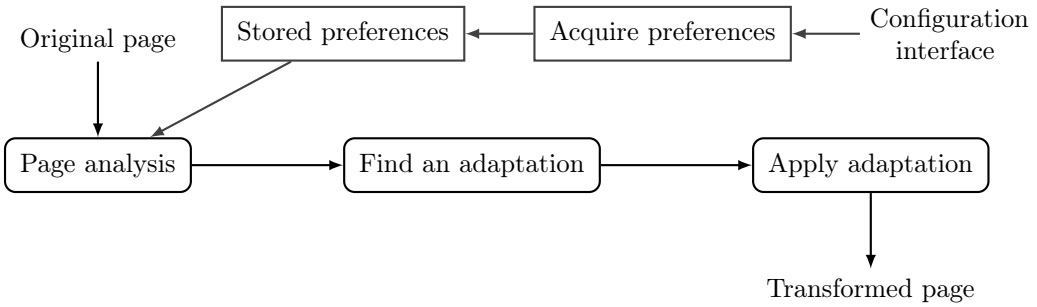


FIGURE 1 – Vue d'ensemble du processus d'adaptation

L'acquisition des préférences de l'utilisateur consiste à lui faire exprimer ses besoins en matière d'adaptation. Ces préférences se traduisent par des règles permettant d'évaluer le niveau de satisfaction d'une page par rapport aux besoins exprimés. Ces préférences sont stockées afin que cette opération ne soit réalisée qu'une seule fois. L'interface reste néanmoins accessible (par un bouton dans la barre d'extension du navigateur) afin de ré-ajuster certains réglages au besoin. Par exemple dans le cas d'une pathologie dégénérative, l'utilisateur aura régulièrement besoin de réajuster sa configuration.

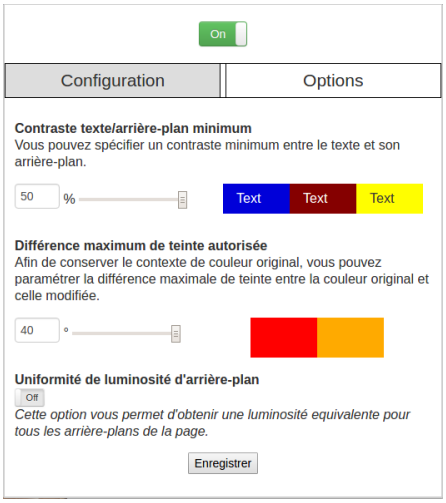


FIGURE 2 – Fenêtre de configuration des préférences

Lorsqu'une page Web est chargée dans le navigateur, celle-ci doit potentiellement être adaptée si elle ne satisfait pas les préférences de l'utilisateur. Pour cela la page Web doit tout d'abord être analysée afin de détecter les différentes structures « visuelles » de la page, c'est-à-dire les structures qui composent la page comme les menus, le fil d'ariane, les entête et pied de page, le contenu principal, etc. De cette manière, il est possible de définir des préférences sur des éléments de haut niveau et non pas uniquement du texte ou des liens. Après l'identification des éléments de la page, ces

derniers sont filtrés pour ne conserver que les éléments qui vont être utiles pour trouver la solution en fonction des préférences définies. Cette étape de filtrage permet de fournir à l'outil de calcul d'une adaptation le moins d'éléments possible pour simplifier la recherche.

Le calcul d'une adaptation fait appel à un algorithme évolutionnaire de type algorithme génétique (NSGA). Dans ces algorithmes, des opérateurs tente de mimer un comportement de la théorie de l'évolution. Nous parlerons donc par exemple d'opération de croisement, de mutation ou de sélection. Il s'agit d'algorithmes dits d'optimisation, c'est-à-dire qu'ils ne parcourent pas l'intégralité des combinaisons possibles pour adapter la page jusqu'à en trouver une bonne (ce qui dans notre cas est impossible au vu du nombre gigantesque de solutions) mais ils évaluent des solutions possibles et tentent de les faire évoluer à travers les opérateurs génétiques.

Une fois qu'une bonne adaptation est calculée, ou que le temps maximum imparti est atteint, l'adaptation renvoyée par l'outil de recherche doit être appliquée sur la page. Cette dernière étape du processus d'adaptation d'une page est réalisée très simplement. Chaque élément passé à l'outil de recherche d'une adaptation est marqué dans la page Web. Ainsi, lorsque l'adaptation est retournée, il est facile de repérer les éléments en question dans l'arbre DOM de la page et de changer leurs propriétés.

3 Découpage et reconnaissance des structures visuelles composant la page

Même si le standard HTML 5 apporte des informations supplémentaires dans le code source par rapport aux version antérieures, la grande majorité des sites Web contiennent au final assez peu d'informations structurelles. Lorsque l'on cherche à améliorer de manière simple le confort de lecture, on va par exemple essayer d'améliorer le contraste entre le texte et son arrière-plan. Ceci peut être réalisé de différentes manières, soit en utilisant des paires de couleurs prédéfinies qui offrent le contraste désiré, soit en effectuant des calculs afin de trouver les couleurs offrant le contraste voulu tout en prenant en compte les couleurs d'origine. Cependant si on veut aller plus loin et améliorer davantage l'accessibilité nous devons faire bien plus qu'améliorer un contraste entre le texte et son arrière-plan. Il pourrait par exemple être intéressant de modifier le menu afin qu'il soit visuellement plus détaché du contenu ou du reste de la page. Le fil d'ariane peut également être difficile à trouver sur certaines pages Web. Le mettre davantage en avant pourrait aider certaines personnes. Tout dépend des habitudes de navigation de chacun. Dans tous les cas, il est important de ne pas se limiter au strict minimum et d'offrir un potentiel d'expression des besoins plus important. Il est donc nécessaire de disposer d'un grand nombre d'informations sur la structuration de la page.

Un outil a été développé en JavaScript sous forme d'extension pour les navigateurs Web afin de s'intégrer très facilement au logiciel de visualisation du contenu Web. Cet outil permet d'effectuer toutes les étape de découpage, de reconnaissance et de filtrage des éléments des pages Web (cf. Figure 3).

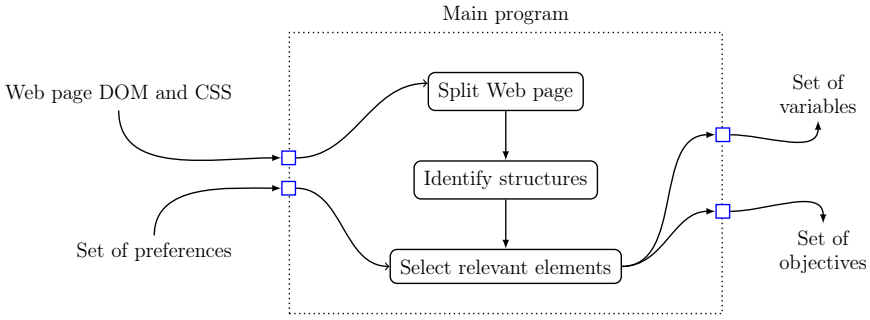


FIGURE 3 – Analyse de la page

Les étapes de découpage et d'identification des structures visuelles découpées se basent uniquement sur les informations disponibles dans le navigateur Web (côté client). Le code HTML et le CSS contiennent les informations nécessaires.

La découpe de la page Web en structure visuelles se fait par plusieurs parcours de l'arbre DOM de la page. Lors de ces parcours, des informations complémentaires (annotations) sont ajoutées dans l'arbre DOM de la page. Ces annotations seront ensuite utilisées pour identifier les structures visuelles extraites. Pour découper la page, nous utilisons par exemple assez fortement la notion de divisibilité d'un nœud. Une série de conditions se basant notamment sur le type de nœud et les nœuds enfants permettent de décider si le nœud courant constitue une entité visuelle unique ou s'il peut être divisé en plusieurs entités visuelles. D'autres notions viennent compléter la divisibilité afin de de ne garder que les éléments utiles et extraire toutes les structures visuelles.

Une fois la liste des structures visuelles extraites, nous pouvons passer à la phase suivante qui consiste à identifier ces structures. Pour ce faire, une série d'heuristiques sont appliquées sur chaque structure visuelle et donnent la probabilité que la structure soit du type associé à l'heuristique. Par exemple la probabilité que telle structure soit un menu, ou telle autre un fil d'ariane. La reconnaissance des structures extraites utilise diverses propriétés. Elle peut utiliser par exemple le type et le nombre d'éléments contenus dans la structure, elle peut également utiliser la position géographique de la structure sur la page mais également la disposition des éléments internes.

La dernière phase consiste à filtrer les éléments extraits. Cette étape a pour but de ne conserver que les éléments HTML (et leurs propriétés) qui sont concernés par les préférences utilisateurs définies. De cette manière en réduisant le nombre d'éléments et propriétés pour chaque élément, on simplifie la résolution du problème d'adaptation.

4 Sélection et fusion des éléments de la page

Les pages Web peuvent facilement contenir plusieurs centaines voire plusieurs milliers d'éléments HTML. Pour chaque élément HTML nous pouvons avoir une multitude de propriétés CSS. De plus les propriétés CSS peuvent avoir des domaines de définition très grands comme celui des couleurs. Essayer de résoudre le problème d'adaptation sur une page complète nous amènerait à ne trouver presque jamais de bonnes adaptations. Il est donc indispensable de se limiter seulement aux éléments ciblés d'une manière ou d'une autre par les préférences utilisateurs. Les autres éléments ne seront pas impactés par l'adaptation il est donc inutile de perdre du temps de calcul à les considérer.

Si l'utilisateur a une pathologie qui entraîne par exemple une photophobie (éblouissement très facile), alors il pourrait utiliser une préférence permettant d'indiquer que l'on souhaite avoir tous les arrière-plans de la page similaires (tous sombres). Si seule cette préférence est définie, il ne sert à rien de s'intéresser aux éléments textuels, tableaux, liens, etc.

Imaginons ensuite qu'un utilisateur définisse également une préférence de contraste minimum entre la couleur du texte et la couleur de son arrière-plan. Il est évident qu'il va falloir récupérer tous les éléments textuels de la page et tous les arrière-plans qui correspondent à ces éléments textuels. La première difficulté consiste à trouver l'arrière-plan qui correspond à un texte. En effet, si visuellement cela semble évident, lorsque l'on regarde le code source cela ne l'est pas toujours. Il arrive souvent que l'arrière-plan d'un texte ne soit pas l'arrière-plan de la boîte qui contient le texte. Si la boîte qui contient le texte a un arrière-plan transparent ou semi-transparent, il faut remonter dans l'arbre jusqu'à trouver une opacité de 100% et récupérer la couleur pour l'associer au texte tout en conservant en mémoire où se trouvait l'arrière-plan pour le modifier ensuite lors de l'adaptation.

Indépendamment de cette phase de récupération des textes et de leurs arrière-plans, la quantité d'éléments récupérés sera souvent très importante. Imaginons un site basique qui a une architecture classique avec un bandeau d'en-tête, un menu, un contenu avec un article, etc. Après récupération, nous aurons un élément textuel pour chaque lien du menu, un élément textuel pour chaque titre, un élément textuel pour chaque paragraphe de l'article, etc. Si on traite ces éléments directement, non seulement on en aura beaucoup à traiter, mais en plus il seront traités de manière indépendante. Autrement dit, des liens qui avaient la même couleur au départ pourraient ne plus avoir la même après l'adaptation. Si des éléments de même nature ont les même propriétés CSS comme la couleur, alors il est souhaitable que l'on conserve cette cohérence après l'adaptation.

Pour conserver la cohérence globale des éléments similaires sur la page d'origine et simplifier la résolution, une phase de fusion des éléments est nécessaire. Nous utilisons donc les propriétés CSS des éléments extraits pour créer des groupes d'éléments de même nature (balise HTML) et d'apparence similaire. La création des groupes prend également en compte l'arrière-plan des éléments de manière à ce qu'un groupe d'éléments textuels ne puisse pas contenir d'éléments se rapportant à des arrière-plans différents. Ensuite, au lieu de traiter des éléments individuellement, l'outil de recherche d'une adaptation utilisera les groupes d'éléments. C'est au moment de l'application de la transformation que l'on reviendra aux éléments HTML à partir des groupes.

Une fois les groupes obtenus, il est possible d'instancier les préférences de l'utilisateur. Par exemple pour la préférence de contraste, pour chaque groupe de textes on aura une instance de la préférence de contraste entre le groupe (d'éléments textuels) et l'arrière-plan qui s'y rapporte. À ce stade on ne parlera plus d'instance de préférence mais d'objectif. Le regroupement permet en plus de réduire le nombre d'éléments à traiter, le nombre d'objectifs définis. L'instanciation d'une préférence utilisateur a pour effet de créer une ou plusieurs variables et un ou plusieurs objectifs. En reprenant la préférence de contraste, pour chaque groupe d'éléments textuels on définira une variable correspondant à la couleur des éléments textuels du groupe et une autre variable correspondant à la couleur de l'arrière-plan de ces éléments textuels. L'ensemble des variables et des objectifs est ensuite passé à l'outil de résolution.

5 Recherche d'une adaptation

La résolution du problème, c'est-à-dire le fait de trouver une adaptation qui satisfasse toutes les préférences de l'utilisateur, est réalisée par un autre outil. Ce dernier n'est pas intégré dans l'extension du navigateur mais est installé sur un serveur distant accessible par le port 80 (http). Cela permet d'alléger la charge processeur incombant à la machine de l'utilisateur et son développement en C++ permet un gain considérable de performances par rapport au langage JavaScript.

Lorsque l'outil d'analyse de la page a terminé l'ensemble des traitements, il émet une requête

au serveur de résolution en donnant l'ensemble des variables et des objectifs. L'outil renverra ensuite l'adaptation calculée.

L'outil de résolution se base sur un algorithme génétique. Plusieurs algorithmes ont été implémentés afin de comparer leur efficacité (NSGA-II, NSGA-III, SPEA-II). NSGA-III est la version qui a permis d'obtenir les meilleurs résultats notamment en terme de temps de calcul et de niveau de satisfaction des objectifs [5]. Cet algorithme fait évoluer une population d'individus (un ensemble de solutions d'adaptation possibles) à travers des opérateurs génétiques. Ces opérateurs permettent par exemple de mélanger des solutions entre elles ou de modifier aléatoirement certaines parties d'une solution dans le but de générer de nouveaux individus (descendance). Ensuite l'opérateur de sélection vient choisir parmi l'ensemble des parents et enfants ceux qui feront partie de la prochaine génération. À chaque génération, les critères d'arrêt sont vérifiés. L'algorithme s'arrête lorsque le temps maximal imparti est atteint (10s) ou bien lorsqu'une bonne solution d'adaptation est trouvée. Une bonne solution d'adaptation est une solution qui satisfait l'ensemble des préférences définies par l'utilisateur. En plus de ces critères d'arrêt, l'ensemble des objectifs doit être évalué pour chaque individu de manière à obtenir son niveau de satisfaction.

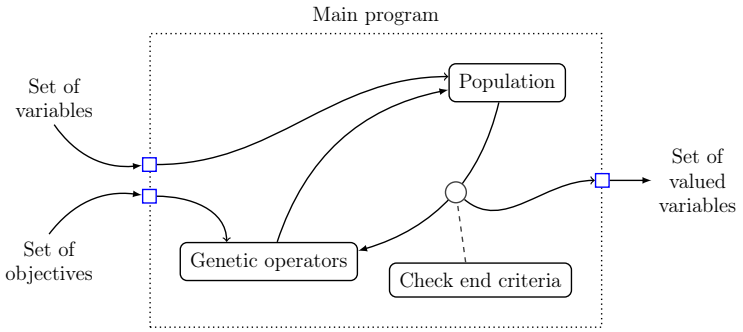
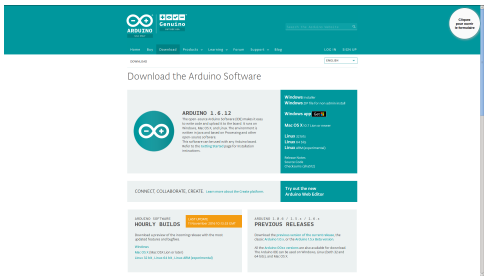


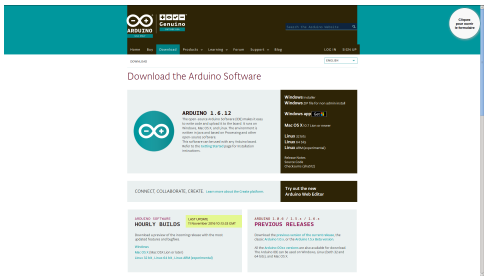
FIGURE 4 – Recherche d'une adaptation

Actuellement l'outil de recherche d'une adaptation utilise un nouvel algorithme génétique. Ce dernier est encore en cours de développement et d'ajustement, mais il offre déjà un gain de performance considérable par rapport à NSGA-III. Cet algorithme reprend les opérateurs de croisement et mutation basiques et récurrents dans les algorithmes génétiques mais apporte un tout nouvel opérateur de sélection basé sur la satisfaction de seuils.

6 Application de la transformation



Site Web Arduino software avant adaptation



Site Web Arduino software après adaptation

Une fois qu'une solution d'adaptation a été retournée par l'outil de résolution, il ne reste plus qu'à l'appliquer sur la page Web. La transformation est relativement simple. Il s'agit, à partir

de chaque variable de la solution, de retrouver l'élément HTML qui lui correspond et de lui affecter la nouvelle valeur de la propriété CSS correspondante. Lorsqu'une variable représente un groupe d'éléments HTML, l'application de la modification est effectuée sur l'ensemble des éléments HTML qui constituent le groupe en question. Cette partie est effectuée dynamiquement en Javascript, par le biais de l'extension du navigateur, sur la page déjà affichée. En effet, le temps que les différents outils s'exécutent, la page Web d'origine telle qu'elle est chargée depuis l'hébergeur est affichée dans la zone de rendu du navigateur. Étant donnés les temps de calcul qui peuvent être longs sur certaines pages, il est donc possible de commencer à découvrir au moins l'architecture de la page ainsi que certaines informations déjà accessibles.

7 Travaux connexes

De nos jours l'adaptation visuelle est souvent réalisée grâce à des outils commerciaux. Ces outils peuvent se trouver directement dans le système d'exploitation, intégré dans le navigateur Web ou sous forme d'extension. Les travaux de recherche autour de ce domaine tentent de prendre de plus en plus en compte les besoins et préférences de l'utilisateur. Des approches comme « color vision deficiency » [15] [17] traitent de déficiences spécifiques, tandis que dans notre cas, nous proposons une approche générique pour toutes les personnes ayant une basse vision. Comme mentionné dans [14], l'architecture des prototypes proposés peut se scinder en outil côté client [14, 15, 4], en proxy [16], host-content-server-side, web page [3] ou en web-service. L'outil peut être destiné à l'utilisateur [15, 14, 12, 16], aux développeurs [12], aux outils d'assistance [10, 11, 9] ou à un usage mixte [4, 3]. Notre outil est destiné à être utilisé côté client et par l'utilisateur final.

L'approche proposée traite différents types d'éléments. Certaines approches ne considèrent que des éléments basiques comme la couleur d'un texte spécifique ou d'un arrière-plan [15, 16], alors que d'autres se focalisent sur les structures et rôles des éléments comme les listes, les menus, les sections ou les titres [10, 11, 9]. Par exemple dans [9], les labels sont assignés aux champs de formulaire par une approche d'optimisation probabiliste. À côté de cela des approches plus avancées proposent leur propre meta-modèle ou langage [12, 10] pour être capables de manipuler des élément de haut niveau d'abstraction. Dans notre cas nous laissons l'utilisateur exprimer ses préférences à différents niveaux d'abstraction, à travers des variables qui décrivent des paires composées d'un élément HTML et d'une propriété potentiellement complexe. Une préférence est n'importe quelle fonction évaluable exprimée sur une ou plusieurs variables et leur domaine de définition. Nous pouvons prendre en compte le design d'origine en utilisant les valeurs d'origine des propriétés des éléments et en définissant les préférences adaptées. Les approches qui utilisent des éléments de haut niveau doivent souvent faire appel à un outil permettant de reconnaître ces structures. Ceci peut être réalisé par l'analyse du code source ou par « pattern recognition » [12].

Au cours du transcodage, les éléments modifiés sont souvent des éléments HTML [10, 16] et des règles CSS [4, 14], Cependant dans certaines conditions il peut aussi s'agir de scripts [14, 15]. Dans [11], le contenu Web est enrichi avec des règles de navigation de contenu (CNR) en introduisant des éléments ARIA. Ceci est destiné à améliorer la navigation avec des lecteurs d'écran. Dans [3], les textes alternatifs sont automatiquement produits soit par reconnaissance de caractères sur les images ou par décodage de l'URL. Les modifications peuvent cibler l'ensemble de la page Web [15], ou une partie spécifique [6, 7, 13]. Dans notre approche nous pouvons modifier une partie ou l'entièreté de la page si nécessaire.

Le profil utilisateur peut se trouver sous la forme d'un ensemble de valeurs attendues pour quelques propriétés comme « font » ou « color » [16, 15, 13]. Le profil utilisateur peut également

être prédéfini selon des catégories d'utilisateurs connues à l'avance comme les différents types de daltonisme [15]. Ce profil utilisateur peut par exemple être saisi à partir d'une interface donnant les valeurs pour les propriétés [16, 14] ou à partir de tests visuels [8]. Dans une approche plus innovante [6, 7, 13], le profil de l'utilisateur est appris à partir de ses actions à travers un algorithme de Q-learning. Dans notre cas nous utilisons actuellement une interface permettant de sélectionner des préférences et leurs principaux paramètres.

En fonction du profil utilisateur, l'étape d'adaptation peut prendre différentes formes. Dans [15], les préférences prédéfinies peuvent être directement appliquées ; dans [13], ce sont les préférences apprises qui sont directement appliquées. Dans le projet Cloud4all [1], les participants travaillent sur une infrastructure inclusive publique globale (GPII) qui permettra l'implémentation facile d'un profil utilisateur pour personnaliser n'importe quel périphérique ou contenu. Dans notre cas, comme les préférences peuvent être conflictuelles, un compromis doit être trouvé. Ceci est réalisé à l'aide de méta-heuristiques.

Certains outils proposent à la fois des adaptations mais aussi des métriques pour juger de la qualité de l'adaptation. Dans notre outil l'utilisation d'algorithmes évolutionnaire fait, qu'à travers les fonctions objectifs, nous pouvons aussi évaluer la qualité des adaptations. Ce sont en quelques sorte des métriques qui guident la recherche d'une adaptation.

8 Conclusion

Alors que les outils d'assistance existants n'offrent que des options « simples » et assez générales, notre projet de recherche veut proposer un outil personnalisable qui adapte les pages Web en fonction des préférences d'une personne spécifique. Un premier outil est disponible et peut être utilisé sur certains navigateurs. Il est constitué des différentes parties décrites dans cet article et utilise la version en cours d'ajustement de l'outil de résolution.

Avec cet outil on peut définir quelques préférences comme notamment celle du contraste ou celle permettant de conserver autant que possible les couleurs d'origine de la page. Les temps de calcul sont limités à une dizaine de secondes (une adaptation de la page est disponible en moins de 10s). Dans le pire des cas, la page ne sera pas ou très peu modifiée. Pour des pages de petite et moyenne taille des améliorations sont possibles en moins de 2 ou 3 secondes. Le projet est encore à un stade expérimental et des progrès peuvent encore être réalisés pour réduire davantage le temps de calcul et améliorer la pertinence des adaptations proposées.

Remerciements Les auteurs remercient Berger Levrault et le Labex NUMEV qui ont soutenu ces travaux.

Références

[1] Cloud4all. http://www.cloud4all.info/. Accessed : 2015-03-25.

[2] Web content accessibility guidelines, 2008.

[3] J. P. Bigham, R. S. Kaminsky, R. E. Ladner, O. M. Danielsson, and G. L. Hempton. Webinsight : : Making web images accessible. In *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility*, Assets '06, pages 181–188, New York, NY, USA, 2006. ACM.

[4] J. P. Bigham and R. E. Ladner. Accessmonkey : a collaborative scripting framework for web users and developers. In *Proceedings of the 2007 international cross-disciplinary conference on Web accessibility (W4A)*, pages 25–34. ACM, 2007.

[5] Y. Bonavero. *Une approche basée sur les préférences et les méta-heuristiques pour améliorer l'accessibilité des pages Web pour les personnes déficientes visuelles. (A meta-heuristic based approach*

for improving Web page accessibility for people with low vision). PhD thesis, University of Montpellier, France, 2015.

[6] S. Ferretti, S. Mirri, C. Prandi, and P. Salomoni. Exploiting reinforcement learning to profile users and personalize web pages. In *IEEE 38th Annual Computer Software and Applications Conference, COMPSAC Workshops 2014, Vasteras, Sweden, July 21-25, 2014*, pages 252–257. IEEE, 2014.

[7] S. Ferretti, S. Mirri, C. Prandi, and P. Salomoni. User centered and context dependent personalization through experiential transcoding. In *Proc. IEEE Consumer Communications and Networking (CCNC 2014), Workshop on Networking Issues in Multimedia Entertainment (NIME'14)*, 2014.

[8] A. Foti and G. Santucci. Increasing web accessibility through an assisted color specification interface for colorblind people. pages 41–48, 2009.

[9] M. A. Islam, Y. Borodin, and I. V. Ramakrishnan. Mixture model based label association techniques for web accessibility. In K. Perlin, M. Czerwinski, and R. Miller, editors, *UIST*, pages 67–76. ACM, 2010.

[10] D. Lunn, S. Bechhofer, and S. Harper. A user evaluation of the sadie transcoder. In S. Harper and A. Barreto, editors, *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2008, Halifax, Nova Scotia, Canada, October 13-15, 2008*, pages 137–144. ACM, 2008.

[11] D. Lunn, S. Harper, and S. Bechhofer. Combining sadie and axsjax to improve the accessibility of web content. In D. Sloan, C. Asakawa, and H. Takagi, editors, *Proceedings of the International Cross-Disciplinary Conference on Web Accessibility, W4A 2009, Madrid, Spain, April 20-21, 2009*, ACM International Conference Proceeding Series, pages 75–78. ACM, 2009.

[12] M. Macías, J. González, and F. Sánchez. On adaptability of Web sites for visually handicapped people. In P. D. Bra, P. Brusilovsky, and R. Conejo, editors, *Proceedings of the second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH)*, volume 2347 of *Lecture Notes in Computer Science*, pages 264–273. Springer, 2002.

[13] S. Mirri, C. Prandi, and P. Salomoni. Experiential adaptation to provide user-centered web content personalization. In *Proc. IARIA Conference on Advances in Human oriented and Personalized Mechanisms, Technologies, and Services (CENTRIC2013)*, pages 31–36, 2013.

[14] S. Mirri, P. Salomoni, C. Prandi, and L. A. Muratori. Gapforape : an augmented browsing system to improve web 2.0 accessibility. *New Review of Hypermedia and Multimedia*, 18(3) :205–229, 2012.

[15] G. Santucci. Vis-a-wis : Improving visual accessibility through automatic web content adaptation. In C. Stephanidis, editor, *Universal Access in Human-Computer Interaction. Applications and Services, 5th International Conference, UAHCI 2009, Held as Part of HCI International 2009, San Diego, CA, USA, July 19-24, 2009. Proceedings, Part III*, volume 5616 of *Lecture Notes in Computer Science*, pages 787–796. Springer, 2009.

[16] B. Tibbitts, S. Crayne, V. Hanson, J. Brezin, C. Swart, and J. Richards. HTML parsing in Java for accessibility transformation. In *Proceedings of XML 2002 – XML Conference and Exposition*, 2002.

[17] L. Troiano, C. Birtolo, and M. Miranda. Adapting palettes to color vision deficiencies by genetic algorithm. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, pages 1065–1072, New York, NY, USA, 2008. ACM.

[18] World Wide Web Consortium, http://www.w3.org/TR/ATAG20/. *Authoring tools Accessibility Guidelines*. Accessed : 2014-11-09.

[19] World Wide Web Consortium, http://www.w3.org/TR/UAAG20/. *User Agent Accessibility Guidelines*. Accessed : 2014-11-09.

[20] World Wide Web Consortium, http://www.w3.org/WAI/intro/aria. *Web Accessibility Initiative - Accessible Rich Internet Applications*. Accessed : 2014-11-09.