Collège Doctoral du Languedoc-Roussillon

THÈSE

Pour obtenir le grade de

Docteur

Délivré par l'**Université de Montpellier**

Préparée au sein de l'école doctorale **I2S***
Et de l'unité de recherche **UMR 5506**

Spécialité: **Informatique**

Présentée par **Yoann Bonavero**
bonavero@lirmm.fr

Une approche basée sur les préférences et les méta-heuristiques pour améliorer l'accessibilité des pages Web pour les personnes déficientes visuelles

Soutenue le 24/11/2015 devant le jury composé de :

M. A. Mille	Pr.	Univ. Claude Bernard Lyon 1	Rapporteur
M. M. Slimane	Pr.	Univ. François Rabelais, Tours	Rapporteur
M. X. Le Pallec	MCF	Univ. de Lille 1	Examinateur
M. J.-C. König	Pr.	Univ. de Montpellier	Examinateur
M. L. Deruelle	Dr.	Berger-Levrault	Invité
Mme M. Huchard	Pr.	Univ. de Montpellier	Directrice de thèse
M. M. Meynard	MCF	Univ. de Montpellier	Co-encadrant

* **I2S :** École doctorale Information Structures Systèmes

Résumé

Lorsque la vue, qui est un important moyen de communication, est altérée, alors l'acquisition de l'information s'en trouve modifiée, dégradée ou limitée. A l'ère du monde numérique, le Web regorge d'informations réparties sur différents sites et mises en forme par les développeurs et designers. De nombreuses pathologies visuelles peuvent entraîner des difficultés dans l'accès à ces informations. Au-delà même de ces informations, l'accès aux outils et services est lui aussi limité. Des difficultés dans la perception des couleurs, des taches dans le champ visuel ou un champ visuel réduit sont tout autant de sources de difficultés. Chaque personne a une vision qui lui est propre. Chez les personnes qui ont une basse vision, les pathologies donnent des évolutions spécifiques chez chacune d'entre elles. De plus les méthodes de compensation acquises sont différentes d'une personne à l'autre. Des outils d'assistance existent depuis de nombreuses années et tentent de répondre aux besoins des personnes ayant une basse vision en proposant des adaptations visuelles. Les principales limites de ces outils résident notamment dans le fait qu'il ne sont pas en capacité de prendre en compte les besoins très spécifiques de chaque personne. Ces travaux de recherche se concentrent donc autour de l'analyse des besoins réels des utilisateurs et de l'élaboration d'une nouvelle approche qui se base sur les préférences personnelles de l'utilisateur. L'objectif final est d'automatiser la transformation des pages Web en fonction des préférences propres à un utilisateur pendant qu'il navigue sur le Web. Divers algorithmes ont été utilisés, notamment des algorithmes évolutionnaires, afin de réaliser des compromis entre les préférences de l'utilisateur et l'apparence originale de la page Web. La thèse développe de manière approfondie les principaux problèmes touchant les personnes en situation de basse vision et des éléments sur les modèles de couleurs et de contrastes. Puis elle présente un langage de modélisation des préférences basé sur la logique, une modélisation du problème comme un problème d'optimisation, des algorithmes de résolution, un démonstrateur, et des expérimentations sur des pages Web réelles.

Abstract

When sight, which is a very important means of communication, is altered, the information acquisition process is also modified, degraded or limited. In today's digital world, the Web contains a wealth of information organized by designers and developers and available on various Websites. Many visual pathologies can lead to difficult access to these sources of information. Beyond just information, access to the tools and services available on the Web is also affected. Color perception disorders, cloud-like white patches or dark areas in a visual field, as well as a reduced visual field all hinder daily web browsing activities. Each person's vision is unique. Several persons affected by the same pathology may even have different perceptions, partly due to the way each one's brain compensates. Several assistive tools have been developed that offer visual adaptation, trying to meet the needs of people with low vision. These tools however often fail to take into account the highly specific needs of each person. The research work presented in this dissertation focuses on analysing the user's actual needs, and on making a new approach to assistive technologies based on the individual user's personal preferences. The final goal is to automatically transform Web pages on the fly according to the specific preferences of each user. Several algorithms have been used, especially evolutionary algorithms, in order to make tradeoffs between the user's preferences and the original appearance of the page. The thesis further develops the main problems encountered by people with low vision and some aspects of color models and contrast relations. Then, we present a preference modeling language based on first-order logic, a modeling of the problem as an optimization problem, some resolution algorithms, a tool prototype and experiments on several real Web pages.

Remerciements

Je profite de ces quelques lignes qui me sont offertes pour remercier l'ensemble des personnes qui ont à leur façon contribué à la concrétisation de ces travaux pour lesquels l'aboutissement était loin d'être évident. En effet, ces travaux apportent une certaine originalité dans les méthodes abordées et rare sont les chercheurs travaillant autour de ces idées dans le monde. Ce n'est donc pas sans une certaine part d'émotion que j'écris ces quelques lignes qui marquent d'une certaine manière la fin d'une étape importante dans la vie d'un chercheur.

C'est donc tout naturellement que mes remerciements se portent tout d'abord sur mes deux encadrants Mme Marianne Huchard et M. Michel Meynard. Tout deux ont réalisé un formidable travail d'encadrement et m'ont apporté d'innombrables connaissances scientifiques pendant ces trois années de recherche même si ces travaux sortent de leurs domaines de prédilection. Ils m'ont fait découvrir le monde de la recherche dans une ambiance de travail très agréable et ont fait part d'une très grande adaptation lorsque j'ai pu faire face à certaines difficultés. Je voudrais également sincèrement remercier Mme. Huchard qui a su percevoir que je cherchais à démarrer des recherches sur un sujet nouveau. C'est donc grâce à elle que j'ai pu commencer à travailler pour passer un concours et obtenir cette thèse dans un domaine pour lequel je me sens concerné.

Mes remerciements se portent à présent sur les membres du jury. Aux deux rapporteurs M. Mohamed Slimane et M. Alain Mille pour le temps qu'ils ont passé à étudier ce mémoire et les remarques, critiques et pistes d'amélioration possibles qui ont été faites. Je remercie l'ensemble des membres du jury pour leur point de vue extérieur qu'ils ont pu exprimer et l'intérêt qu'ils ont pour ces travaux de recherche et notamment ceux qui souhaitent s'investir davantage dans la suite de ces travaux.

Je tiens également à remercier M. Laurent Deruelle et M. Laurent Smadja avec qui j'ai principalement été en contact et qui représentent l'entreprise Berger-Levrault. Ce partenaire industriel a soutenu ces travaux de recherche sur ces deux dernières années et souhaite continuer à soutenir les travaux futurs.

Pour finir, je voudrais remercier toutes les autres personnes que je ne peux citer ici mais qui, je l'espère, se sentiront concernées. Toutes les personnes qui m'ont soutenu de près ou de loin dans les diverses phases plus ou moins faciles que j'ai pu traverser tout au long de ces trois années.

Table des matières

1 Introduction **18**

1.1 L'accessibilité et le numérique 19
1.2 Le handicap 20
1.3 Cadre du sujet et axes directeurs 22

2 Notions complémentaires requises **26**

2.1 L'œil humain et son fonctionnement 27
2.2 L'acuité visuelle 28
2.3 La déficience visuelle 29
2.3.1 Troubles visuels dus à des pathologies 29
2.3.2 Regroupement des pathologies visuelles 35
2.4 Les couleurs et leurs propriétés 36
2.4.1 Organisation des couleurs 36
2.4.2 Des propriétés diverses 37
2.5 Conclusion 41

3 État de l'art **42**

3.1 Le Web dans sa forme visuelle 44
3.2 La normalisation 45
3.2.1 Domaine d'application et cadre juridique 46
3.2.2 Les éléments sur lesquels portent les normes et recommandations . 46
3.2.3 Le contenu Web 46
3.2.4 Les outils de production 47
3.2.5 Les outils de consultation 48
3.2.6 Un exemple plus concret 48
3.2.7 Les limites 49
3.3 Les technologies d'assistance 49
3.3.1 Les *magnifiers* d'écran 49
3.3.2 Intégration dans les systèmes d'exploitation 53
3.3.3 Les Outils de gestion des styles 54
3.4 Adaptation de pages Web et préférences utilisateur 56
3.4.1 Architecture de transformation, destinataires et but 56
3.4.2 Éléments traités, reconnaissance automatique, cible d'application, partie transformée et conservation du contexte originel 57
3.4.3 Profil utilisateur : langage de définition, acquisition, déficience spécifique visée et définition a priori 59
3.4.4 Méthode d'adaptation 60

3.4.5 Évaluation, métriques et utilisateurs . . . 60
3.4.6 Démonstrateur et corpus de données fourni . . . 61
3.5 Conclusion . . . 61

4 Problématiques de visualisation des interfaces Web 64

4.1 Problèmes sur la page d'origine . . . 65
4.1.1 Éblouissements (brillance importante) . . . 65
4.1.2 Contraste faible . . . 66
4.1.3 Texte au format image . . . 67
4.1.4 Images d'arrière-plan . . . 68
4.1.5 Ordre du contenu . . . 69
4.1.6 Le daltonisme . . . 69
4.2 Utilisation des technologies d'assistance . . . 70
4.2.1 Éblouissements (brillance importante) . . . 71
4.2.2 Contraste faible . . . 72
4.2.3 Images d'arrière-plan . . . 72
4.2.4 Ordre du contenu . . . 74
4.2.5 Le daltonisme . . . 74
4.3 Conclusion . . . 74

5 Analyse des besoins en vue de l'acquisition des préférences de l'utilisateur 76

5.1 Introduction . . . 77
5.2 Types de besoins et leur expression . . . 78
5.3 Analyse des réponses . . . 79
5.4 Synthèse des réponses et observations . . . 81
5.5 Souhaits et préférences . . . 82
5.5.1 Acquisition par l'exemple . . . 83
5.5.2 Acquisition par apprentissage . . . 84
5.6 Conclusion . . . 85

6 Recherche d'une adaptation 86

6.1 Approche exacte - La théorie des préférences . . . 87
6.2 Syntaxe du langage des préférences . . . 88
6.3 Sémantique du langage des préférences . . . 89
6.3.1 Méthodes de résolution . . . 90
6.3.2 Limites principales liées au contexte . . . 92
6.4 Les algorithmes évolutionaires . . . 93
6.4.1 L'algorithme NSGA - Concepts fondamentaux . . . 94
6.4.2 La variante NSGA-II . . . 96
6.4.3 La variante NSGA-III . . . 97
6.4.4 Le tri par fronts de non domination . . . 98
6.4.5 Les opérateurs génétiques . . . 100
6.4.6 Les fonctions objectif . . . 108
6.5 Conclusion . . . 113

7 Schéma global du démonstrateur **114**
7.1 Sélection des données depuis la page Web . . . 115
7.2 Données en entrée . . . 116
7.3 Données en sortie et transformation de la page Web . . . 118
7.4 Architecture générale . . . 118
7.5 Les différents paramètres . . . 119
7.6 Exemples d'adaptations . . . 119
7.7 Conclusion . . . 121

8 Expérimentations **122**
8.1 Introduction . . . 123
8.2 Approches exactes . . . 123
8.3 Algorithmes génétiques . . . 125
8.3.1 Mise en œuvre . . . 126
8.3.2 Corpus de pages Web réelles utilisé . . . 131
8.3.3 Résultats . . . 132
8.3.4 Impact du choix des paramètres . . . 142
8.4 Conclusion . . . 145

9 Conclusion et perspectives **146**
9.1 Rappel du contexte . . . 146
9.2 Contributions . . . 147
9.3 Perspectives ouvertes . . . 148
9.3.1 Acquisition des préférences . . . 148
9.3.2 Méthodes de résolution . . . 149
9.3.3 Validation par les utilisateurs concernés . . . 150

Références **156**

A Pages Web pour l'expérimentation **157**
A.1 Legibase (Berger-Levrault) . . . 158
A.2 National Federation of the Blind (NFB) . . . 159
A.3 GDArchitect . . . 160
A.4 My Account (Berger-Levrault) . . . 161
A.5 Parempuyre . . . 162
A.6 Godaddy . . . 163
A.7 FamilyPortal (Berger-Levrault) . . . 164
A.8 UBOFormation . . . 165
A.9 UBOLEA . . . 166
A.10 ReasonFrance . . . 167

B Extraction des éléments des pages Web **168**
B.1 Legibase (Berger-Levrault) . . . 169
B.2 National Federation of the Blind (NFB) . . . 171
B.3 GDArchitect . . . 173
B.4 My Account (Berger-Levrault) . . . 175
B.5 Parempuyre . . . 177
B.6 Godaddy . . . 179
B.7 FamilyPortal (Berger-Levrault) . . . 181
B.8 UBOFormation . . . 183
B.9 UBOLEA . . . 185
B.10 ReasonFrance . . . 187

Table des figures

2.1 L'œil humain 27
2.2 Impact de la DMLA 31
2.3 Impact de la cataracte 31
2.4 Impact de la rétinopathie diabétique 32
2.5 Impact du glaucome 33
2.6 Impact de la rétinite pigmentaire 33
2.7 Impact de névrite optique 34
2.8 Impact de l'Hémianopsie homonyme 35
2.9 Classification de l'ensemble des pathologies support 35
2.10 Disque chromatique de saturation par synthèse additive 36
2.11 Différence de brillance pour les trois couleurs primaires R, v et B 38
2.12 Exemples de contrastes de brillance 39
2.13 Contrastes simultanés 40
2.14 Perception des dimensions selon la couleur 41

3.1 Modes de zoom avec les agrandisseur d'écran 50
3.2 Filtre d'inversion de couleur sur du texte 51
3.3 Amélioration du pointeur et du curseur 52
3.4 Lecture du texte dans un lecteur externe 53
3.5 Changement de couleurs par la feuille de style (Google Chrome) 55

4.1 Effect d'éblouissement par l'arrière-plan (lettre gauche : pas d'éblouissement, lettre droite : éblouissement) 65
4.2 Exemples de contrastes faibles 66
4.3 École nationale de Montpellier (Menu) 66
4.4 École nationale de Paris (accueil) 67
4.5 Contraste non uniforme (texte sur image) 68
4.6 Effet de la protanopie (type de daltonisme) 70
4.7 Brillances opposées et filtre d'inversion de couleurs 71
4.8 Application de filtres 72
4.9 Perte d'informations par la désactivation des images d'arrière-plan 73
4.10 Perte d'éléments d'interaction par la désactivation des images d'arrière-plan 73

5.1 Du besoin d'accès à l'information aux souhaits 82
5.2 Exemple de souhait vers préférence 83

6.1 Schéma général d'une itération des algorithmes NSGA-II and NSGA-III . . 95
6.2 NSGA-III hyperplan, points et lignes de références 98
6.3 Opérateur de croisement donnant un seul enfant 101

6.4 Opérateur de croisement donnant deux enfants 102
6.5 Opérateur de mutation à base aléatoire 103
6.6 NSGA-II et NSGA-III sélection des parents 106
6.7 Exemple d'agrégation des fonctions objectif 112

7.1 Sélection des éléments sur la page Parempuyre 116
7.2 Exemples de variables pour le page Web Parempuyre 117
7.3 Exemple d'objectifs pour le la page Web Parempuyre 117
7.4 Exemples de valeurs de retour associées aux variables 118
7.5 Schéma résumé du démonstrateur 118
7.6 Transformation (contraste min et conserv. contexte couleur d'origine) - *NFB* 120
7.7 Transformation (contraste min et conserv. contexte couleur d'origine) - *Godaddy* 120

8.1 Amélioration apportée par NSGA-III sur les pourcentages d'objectifs satisfaits et le temps de calcul 141
8.2 Temps d'exécution / Génération en fonction de la taille de population (Exemple : Godaddy S_1) 143
8.3 Temps d'exécution en fonction de (1 - taux de mutation) (Godaddy S_1) . . 144

A.1 Page du site Web *Legibase* 158
A.2 Page du site Web NFB 159
A.3 Page du site Web *GDArchitect* 160
A.4 Page du site Web de Berger-Levrault 161
A.5 Page du site Web Parempuyre 162
A.6 Page du site Web *Godaddy* 163
A.7 Page du site Web de Berger-Levrault *Portail famille* 164
A.8 Page du site Web UBO Formations 165
A.9 Page du site Web UBO Licence LEA 166
A.10 Page du site Web Reason France (forum) 167

B.1 Page du site Web *Legibase* 169
B.2 Page du site Web NFB 171
B.3 Page du site Web *GDArchitect* 173
B.4 Page du site Web de Berger-Levrault 175
B.5 Page du site Web 177
B.6 Page du site Web *Godaddy* 179
B.7 Page du site Web de Berger-Levrault *Portail famille* 181
B.8 Page du site Web UBO Formations 183
B.9 Page du site Web UBO Licence LEA 185
B.10 Page du site Web Reason France (forum) 187

Liste des tableaux

3.1 Comparaison - Architecture et destinataires 57
3.2 Comparaison - Eléments étudiés et modifiés dans les pages 58
3.3 Comparaison - Profil de l'utilisateur 59
3.4 Comparaison - Évaluations 61

6.1 Véhicules colorés 89

8.1 Algorithmes prefs - Quantifications et estimation du temps d'exécution . . 124
8.2 Paramètres communs de configuration pour NSGA-II et NSGA-III 126
8.3 Paramètre de configuration propre à NSGA-II 127
8.4 Paramètres de configuration propres à NSGA-III 127
8.5 Configuration des objectifs utilisateur 129
8.6 NSGA-II sur « Godaddy » **sans** agrégation des ensembles d'objectifs . . . 132
8.7 NSGA-II « Godaddy » avec ensembles d'objectifs agrégés 133
8.8 NSGA-III « Godaddy » avec **sans** agrégation des ensembles d'objectifs . . 134
8.9 NSGA-III « Godaddy » avec agrégation des ensembles d'objectifs 135
8.10 Caractéristiques des pages Web étudiées 137
8.11 Comparaison globale (S_1 et S_3) sur plusieurs pages réelles (Temps/ exécutions $\geq$ 10s / satisfaction des exécution $\geq$ 10s) 137
8.12 Comparaison globale (S_2 et S_4) sur plusieurs pages réelles (Temps/ exécutions $\geq$ 10s / satisfaction des exécutions $\geq$ 10s 138
8.13 Comparaison globale (S_{1agr} et S_{3agr}) sur plusieurs pages réelles (Temps/ exécutions $\geq$ 10s / satisfaction des exécutions $\geq$ 10s) 139
8.14 Comparaison globale (S_{2agr} et S_{4agr}) sur plusieurs pages réelles (Temps/ exécutions $\geq$ 10s / satisfaction des exécutions $\geq$ 10s 139

Liste des Algorithmes

1 Sélection des parents - concepts communs à NSGA-II et NSGA-III 95
2 Tri rapide en fronts de non domination 99
3 Croisement donnant un enfant 101
4 Croisement générant deux enfants 102
5 Opérateur de mutation d'un individu 103
6 Sélection par tournois binaires 104
7 Retourne le meilleur des deux individus 104
8 Calcul de la distance de foule 105
9 Sélection environnementale 107
10 Fonction objectif - Uniformité des brillances de couleur 109
11 Fonction objectif - Contraste de brillance minimum 110
12 Teinte à partir d'une couleur sRGB 111
13 Fonction objectif - Distance de couleur maximale 111
14 Génération des relations de préférences (ensembles comparatifs) 124

Glossaire

API *Application Programming Interface*. Constitue un ensemble de fonctions/procédures pour l'utilisation simplifiée d'éléments de plus bas niveau. 16

ATAG *Authoring Tool Accessibility Guidelines*. Un ensemble de standards et recommandations d'accessibilité relatif aux outil permettant la génération (édition) de contenu Web. 16, 47, 48

BEV *Bandes d'Éveil de Vigilance*. bandes en relief permettant le guidage, le suivi et la prévention d'un danger. 16, 19

ERP *Établissement recevant du public*. 16, 22

NSGA *Non-dominated Sorting Genetic Algorithm*. Un algorithme génétique multi-objectif permettant de traiter des problèmes d'optimisation. 16, 86, 94, 97, 99, 113, 114, 118, 128, 134, 145

PMR *Personne à Mobilité Réduite*. 16, 45

RGAA *Référentiel Général d'Accessibilité pour les Administrations*. Un ensemble de standards et recommandations pour la mise en accessibilité des pages Web dépendantes des services de l'état et des collectivités territoriales qui en dépendent. 16

signe Trouble observable par un individu externe. 16

symptôme Trouble ressenti par un individu. 16, 30, 33, 35

technologie d'assistance (ou outil d'assistance) Divers outils logiciels, commerciaux ou non, permettant l'adaptation d'un poste de travail. 16, 42, 45, 47, 49, 62, 70, 71

TIC *Technologies de l'Information et de la Communication*. 16, 22, 45

UAAG *User Agent Accessibility Guidelines*. Un ensemble de standards et recommandations pour la mise en accessibilité des applications de navigation (navigateur Web) ou de visualisation de pages Web. 16, 48

WCAG *Web Content Accessibility Guideline*. Un ensemble de standards et recommandations pour la mise en accessibilité des contenus Web. 16, 46, 47, 67, 68

Chapitre 1

Introduction

Le 21^{e} siècle tel que nous le connaissons a vu émerger, depuis quelques années, de nombreuses technologies nouvelles. Ces technologies permettent notamment un accès à l'information et à la communication quel que soit le lieu (ou presque) dans lequel on se trouve. Elles favorisent constamment le développement d'une société numérique. Les terminaux mobiles tendent à remplacer les plate-formes fixes comme les ordinateurs « de bureau » tels que nous les connaissons. Bien sûr il ne s'agit que d'une tendance, certains corps de métiers continueront à avoir besoin d'utiliser des plate-formes fixes. La mobilité (qui se mesure par des caractéristiques telles que la taille, la masse, l'encombrement ou encore l'autonomie) ainsi que l'ergonomie et le type d'interactions avec le périphérique font partie intégrante des axes directeurs majeurs dans le processus de conception des nouveaux outils.

Les *smartphones* et autres périphériques mobiles remportent en ce début de siècle un vif succès auprès de la population. Mais ces terminaux à eux seuls ne suffisent pas à expliquer l'engouement du public pour ces téléphones et tablettes. La notion d'hyperconnexion offre d'innombrables possibilités de communication, d'accès à l'information et d'utilisation de services. C'est cette notion d'hyperconnexion qui leur donne tout leur intérêt. L'accès aux journaux, à la météo, aux réseaux sociaux, la consultation de comptes bancaires ou des horaires des transports en commun, la recherche d'un lieu-dit ou d'un commerce à proximité ou bien la navigation assistée sont autant de services devenus tellement courants qu'ils en deviennent presque incontournables. Les machines fixes n'en sont pas pour autant complètement abandonnées, elle possèdent des avantages non négligeables en terme de coût, de performances, d'utilisabilité ou encore de flexibilité et parfois même de confort. Les machines fixes ont également comme point commun avec les terminaux mobiles cette notion d'hyperconnexion. Les services disponibles sur les périphériques mobiles le sont aussi sur les machines fixes parce qu'il s'agit de services dits « en ligne », c'est-à-dire qu'ils utilisent le réseau Internet pour se propager n'importe où. L'utilisation des protocoles du Web leur permettent d'accroître considérablement leur interopérabilité.

Le principal point commun à tous ces outils se trouve dans la transmission de l'information depuis des dispositifs tels que l'ordinateur, le téléphone, ou la tablette vers l'utilisateur. Si la transmission de l'information de l'utilisateur vers la machine se fait à partir d'interfaces diverses comme le clavier ou la souris pour les ordinateurs de bureau, les glaces tactiles et/ou claviers virtuels pour les terminaux mobiles, l'acquisition de l'informa-

tion par l'utilisateur depuis le terminal passe essentiellement par le visuel. L'information peut se trouver sous la forme d'images portant une signification (icônes, logo), sous la forme de texte plus ou moins organisé, sous la forme de couleurs, de photos, de vidéos, de schémas ou de clips et de bruitages audio. Cette liste n'est pas exhaustive mais couvre tout de même la plupart des supports de communication et il y apparaît très clairement que la fonction visuelle est prédominante dans la transmission d'informations. Ce modèle de communication se retrouve de manière générale chez l'homme, que ce soit lors de la phase d'adaptation d'un nourrisson ou bien à l'âge adulte. Lors d'un dialogue on pourra percevoir par des signes visuels l'intérêt ou le désintérêt d'une personne, s'il comprend ou non, quel est son degré d'attention, s'il acquiesce ou non. On estime à environ 80% la part d'information transmise et recueillie par le canal visuel dans la vie quotidienne.

Que se passe-t-il maintenant si on altère ce canal de communication ? Si on modifie la perception des informations visuelles ? Comment peut-on compenser les informations perdues ou erronées ? Ces travaux de recherche ont pour but d'améliorer l'accès à l'information des personnes atteintes d'un déficit visuel lorsqu'elles font face aux nouvelles technologies de l'information et de la communication comme le Web. Dans le cadre de ces travaux de recherche, nous étudierons les technologies d'assistance existant à ce jour et leurs bénéfices sur la perception de l'information. Notre objectif principal est de proposer une nouvelle approche, qui soit réellement personnalisée et personnalisable, qui permette d'améliorer la qualité de l'information transmise sous forme visuelle et ce, en préservant autant que possible le message visuel initial. On parle alors véritablement d'accessibilité à l'information.

1.1 L'accessibilité et le numérique

On peut définir l'accessibilité, dans son acception la plus générale, comme suit :

> *« L'accessibilité désigne le caractère possible de la liberté de déplacement dans l'espace, d'utilisation d'outils, et de compréhension. À ne pas confondre avec l'ergonomie et l'utilisabilité. Un des principaux aspects de l'accessibilité est spécifique aux handicaps, mais d'autres aspects existent. Ce terme est aussi utilisé dans la Convention relative aux droits des personnes handicapées. »*
>
> Wikipedia

Cette définition très générale de l'accessibilité s'applique aussi bien à la voirie, aux transports, au cadre bâti, qu'au monde virtuel du numérique. Elle cherche à proposer dans tous ces domaines une conception architecturale permettant à toute personne de se déplacer librement et de manière autonome indépendamment de ses aptitudes physiques ou mentales. Dans le cadre de la voirie, on notera par exemple les abaissements de trottoirs au niveau des passages piétons et des sorties de garages. On trouve aussi de plus en plus de bandes d'éveil de vigilance BEV pour indiquer les passages piétons et les escaliers. Le contraste des marquages et signalétiques fait aussi partie des normes concernant la voirie. Ainsi les panneaux routiers et les panneaux d'information devront être contrastés

dans leurs contenus et de manière générale l'information à véhiculer doit être contrastée avec son environnement immédiat. Pour les transports on va chercher le "tout à zéro", à supprimer les ressauts et marches, et améliorer les contraintes.

L'accessibilité numérique, quant à elle, peut se définir comme la mise à disposition de ressources numériques à toute personne et ce, indépendamment de son matériel, de son infrastructure réseau, de ses logiciels, de son emplacement géographique ou de ses aptitudes physiques et/ou mentales (cf. décret *2009-546* du 14 mai 2009 pris en application de l'article 47 de la loi *2005-102* du 11 février 2005 relative à l'égalité des droits et des chances, la participation et la citoyenneté des personnes handicapées). Cette loi et sa mise en application par le biais du décret concernent ici exclusivement le secteur public en lui proposant un référentiel général d'accessibilité et les ressources numériques fournies par ses services.

Dans ce contexte, l'accessibilité numérique couvre par exemple :

— La téléphonie mobile à partir de la troisième génération. Cela concerne la conception matérielle, logicielle et de services.

— La télévision et la radio numérique (TNT, RNT...). Cela concerne la conception du matériel et des services mais aussi la compatibilité et les normes.

— Les communications numériques à large bande.

— Tous les usages qui peuvent être faits sur des systèmes ou réseaux qu'il soient d'ordre privé ou professionnel.

— etc.

La prise en compte de ces différents aspects, dans les phases de conception et de développement de matériels, applications ou services, a été dans le passé qualifiée de « peu intéressante » et de « perte de temps », simplement parce que ceux-ci ne visaient qu'un public assez peu important et bien trop ciblé. Depuis quelques années, ces aspects reviennent sur le devant de la scène et sont de plus en plus mis en avant et pris en compte. Contrairement aux idées reçues du passé, l'accessibilité bénéficierait à un public très large au-delà même des personnes porteuses d'un handicap. Prendre en compte la déficience auditive pourra bénéficier aux utilisateurs se trouvant dans un milieu bruyant ou dont le système audio est désactivé ou inexistant sur le poste de travail. De la même manière, prendre en compte la déficience visuelle pourra améliorer le confort des utilisateurs se trouvant dans des conditions d'éclairage peu favorables comme une forte exposition au soleil. Des décisions ont été prises par plusieurs pays et l'Europe considère l'accessibilité numérique comme une obligation citoyenne.

Pour résumer, l'accessibilité numérique signifie que tous les contenus doivent être consultables et utilisables par tous.

1.2 Le handicap

Le terme de handicap n'a pas toujours eu une connotation péjorative. Il a été utilisé pour la première fois en 1827. Ce mot vient de l'évolution de trois mots signifiant main

dans le chapeau « hand in cap ». Autrefois, lors d'un troc entre deux personnes, si les objets échangés n'avaient pas la même valeur alors il fallait rétablir l'équité entre les deux objets grâce à de l'argent. De cette manière, la personne recevant un objet d'une valeur plus importante que celle de l'objet donné devait compenser en donnant de l'argent dans un chapeau. Ainsi le troc devenait équitable. C'est ensuite au domaine sportif, qu'au fil du temps et à force de transformations, cette expression s'est appliquée. Elle s'appliquait notamment à cette époque-là aux courses hippiques et désignait le fait de favoriser ou de défavoriser les participants en fonction de leurs performances dans le but de donner sa chance à chacun d'entre eux. Historiquement, dans le domaine de la santé, le handicap se définissait par opposition à la maladie. Lorsqu'une personne était atteinte d'une affection, tant qu'elle était prise en charge par un service médical alors elle était considérée comme simplement malade. Mais dès lors qu'elle n'était plus suivie et que l'affection était reconnue comme incurable, elle était considérée comme handicapée.

C'est un peu plus d'un siècle et demi plus tard, en 1980, qu'un Britannique du nom de Philip Wood a radicalement changé le sens du mot handicap en l'associant à un désavantage plutôt qu'à une notion d'égalité. Cet handicap désigne donc une difficulté pour un individu à l'accomplissement d'une tâche sociale de manière « normale ». Cette difficulté résulte d'une incapacité due à une déficience qu'elle soit temporaire, durable ou définitive. C'est dans cette définition du handicap que l'on se trouve aujourd'hui même si elle tend à évoluer pour prendre davantage en compte les facteurs sociaux et inversement à inhiber les facteurs fonctionnels.

Les facteurs environnementaux ne sont pas négligeables et tendent à être davantage pris en compte. Dans ce sens l'OMS [1] introduit une nouvelle définition du handicap. Elle s'appuie en partie sur la Classification Internationale du Fonctionnement et la Classification Internationale du Handicap et de la santé. Elle définit alors le handicap comme étant l'association d'une déficience avec une situation de la vie quotidienne. La déficience représente ici l'ensemble des affections et problèmes de santé (pathologies, maladies, blessures) et la situation sociale quant à elle varie en fonction du contexte environnemental actuel.

Le site officiel de l'OMS définit les handicaps de cette manière : « Les handicaps sont un terme général qui qualifie des incapacités, une limitation des actes de la vie quotidienne et une participation restreinte à la vie sociale » et mentionne également : « Le handicap n'est pas simplement un problème de santé. Il s'agit d'un phénomène complexe qui découle de l'interaction entre les caractéristiques corporelles d'une personne et les caractéristiques de la société où elle vit. Pour surmonter les difficultés auxquelles les personnes handicapées sont confrontées, des interventions destinées à lever les obstacles environnementaux et sociaux sont nécessaires ».

Il existe aujourd'hui plusieurs définitions publiées par des organismes différents. Cependant celles-ci se rejoignent et partagent les même concepts.

1. Organisation Mondiale de la Santé

> *« Constitue un handicap, au sens de la présente loi, toute limitation d'activité ou restriction de participation à la vie en société subie dans son environnement par une personne en raison d'une altération substantielle, durable ou définitive d'une ou plusieurs fonctions physiques, sensorielles, mentales, cognitives ou psychiques, d'un polyhandicap ou d'un trouble de santé invalidant. »*
>
> Loi *n ° 2005-102 du 11 février 2005*

D'après ces définitions, le handicap regroupe un grand nombre d'affections de provenances diverses. Le handicap peut dépendre du contexte dans lequel la personne se trouve, par exemple, la myopie, qui est un défaut visuel très courant, peut être considérée comme un handicap dès lors qu'elle empêche la réalisation correcte d'une tâche commune.

On recense, d'après *le rapport mondial sur le handicap* de l'OMS [OMSf], plus d'un milliard de personnes vivant avec un handicap sous une forme ou une autre, dont plus de 200 millions d'entre elles ont de très grandes difficultés fonctionnelles. Le nombre de personnes en situation de handicap ne cesse d'augmenter, notamment avec la croissance de la population mondiale. Beaucoup d'enfants naissent avec un handicap, mais ce n'est pas la totalité des personnes adultes reconnues handicapées. Les accidents de la route, les conflits (guerres), les opérations médicales à risque, ou encore les maladies peuvent donner lieu à un ou plusieurs types de handicap : moteur, sensoriel, mental, cognitif ou psychique.

1.3 Cadre du sujet et axes directeurs

L'accessibilité est un terme englobant un vaste domaine incluant celui du transport, de la voirie, des ERP, du numérique, etc. Les travaux de recherche que nous présentons ici se focalisent sur les nouvelles technologies de l'information et de la communication (TIC) et sur la vision. La limite est encore plus précise puisque nous nous restreindrons uniquement à une sous-partie du monde des nouvelles technologies, celle du Web et son accessibilité. Ce choix provient à la fois d'une nécessité de délimiter le champ d'action à l'intérieur même des nouvelles technologies, mais aussi d'un manque à combler important des technologies d'assistance opérant dans ce domaine auprès des personnes atteintes par une déficience.

Dans ce cadre, l'utilisation du Braille est l'un des premiers dispositifs qui vient à l'esprit pour les personnes déficientes visuelles. Le projet NatBraille [LIR15], par exemple, propose un moteur de transcription/détranscription universel Braille intégrable dans des applications et utilisable sans connaissance préalable du Braille. Le projet s'adresse à la (dé)transcription de tous les documents, que ce soit du texte mais aussi des formules de mathématiques, de chimie, etc. Ici, nous ne développerons pas les approches d'assistance permettant aux personnes atteintes de cécité totale d'acquérir ou de produire de l'information, nous nous concentrerons sur le domaine de la basse vision.

Nous nous limiterons à la visualisation, à la perception et à l'acquisition des informations et ressources sur les pages Web et à l'utilisation d'applications Web par des personnes dont la vision est altérée suite à diverses pathologies visuelles.

L'orientation choisie vers des approches personnalisées nous pousse à nous rapprocher des besoins réels et concrets de chaque utilisateur potentiel. Une première phase impor-

tante consiste donc à définir, apprendre et maîtriser les besoins potentiels des utilisateurs, ainsi que les points sur lesquels il est possible de jouer pour améliorer leur confort visuel et la qualité de leurs interactions. Cette connaissance des besoins des utilisateurs est le point clef pour apporter des améliorations permettant de concevoir le schéma global d'une interface de configuration. Cette interface devra permettre aux utilisateurs, quels que soient leurs déficiences et leurs besoins, de les renseigner. La dernière phase consiste donc en la manipulation des besoins recueillis afin d'apporter une amélioration personnalisée. De manière générale, nous allons proposer, à l'aide de technologies d'optimisation (exactes ou approchées) d'adapter l'affichage des pages Web en fonction des besoins spécifiques d'un utilisateur et ce en préservant autant que possible l'apparence initiale.

Thèse défendue

« Il est possible, grâce à des méthodes d'optimisation, de réaliser un compromis entre les besoins des utilisateurs et le message visuel initial de la page Web pour permettre à un utilisateur en situation de basse vision de profiter au maximum de ses capacités visuelles et de la page web d'origine »

- ⊙ Il est possible d'acquérir les besoins spécifiques d'un utilisateur ayant une basse vision
- ⊙ Il est possible de les instancier sur une page (automatiquement ou pas) sous forme de « contraintes » quantifiables par des métriques (les fonctions objectif)
- ⊙ Il est possible de trouver une adaptation personnalisée de la page pour des personnes ayant une basse vision à l'aide d'une méthode méta-heuristique et ce, au moins pour des pages de petite ou moyenne taille
- ⊙ Il est possible de transformer une page avec la solution obtenue grâce à la méta-heuristique

Afin de répondre aux différents points listés ci-dessus, divers domaines de recherche seront abordés. Le chapitre 2 donne tout un ensemble de connaissances utiles pour mieux comprendre le contexte ainsi que les propriétés utilisées par la suite comme celles des couleurs. Une partie concerne la perception humaine et l'impact des pathologies visuelles sur le champ visuel. Le chapitre suivant (chapitre 3) fait le tour des technologies d'assistance existantes qui tentent d'améliorer l'accessibilité des pages Web pour les personnes ayant

une basse vision. Il parcourt également plusieurs travaux de recherche autour de la transformation de pages Web. Les points clés et limites de ces travaux sont montrés et nous décrivons comment notre approche essaye de dépasser ces limites. Le chapitre 4 quant à lui vient apporter un complément sur l'utilisation des technologies d'assistance existantes. Il permet de mettre en avant des difficultés précises rencontrées par des utilisateurs ayant une basse vision sur des pages Web, et comment les outils d'assistance interviennent pour améliorer l'accessibilité. Dans ce chapitre, plusieurs limites sont mises en avant selon le type de problème rencontré sur la page initiale.

Le chapitre 5 se base sur le chapitre précédent pour réaliser une première analyse des besoins que peut avoir un utilisateur malvoyant, les besoins spécifiques d'un utilisateur étant au cœur de notre approche. Nous expliquons dans ce chapitre comment il est possible de passer des besoins d'un utilisateur vers des préférences qui conditionneront l'adaptation de la page. À partir de ces préférences un processus de recherche d'une adaptation est décrit dans le chapitre 6. Plusieurs méthodes de recherche d'une bonne adaptation sont parcourues. Le chapitre 7 décrit l'implémentation des méthodes de résolution, utilisées pour rechercher une bonne adaptation, dans un démonstrateur. Ce démonstrateur est ensuite utilisé pour réaliser des expérimentations sur des pages réelles avec divers ensembles de préférences. Le chapitre 8 se consacre donc à ces expérimentations permettant entre autres de valider l'approche et de comparer les résultats obtenus par plusieurs algorithmes.

Le chapitre 9 vient donner les conclusions générales sur le projet de recherche et les différentes contributions apportées tout au long des ces travaux. Il ouvre également un certain nombre de perspectives qui seront étudiées dans des futurs travaux.

Chapitre 2

Notions complémentaires requises

Préambule

Dans ce chapitre, nous abordons des notions fondamentales sur la biologie humaine (l'œil) qui seront utilisées par ces travaux de recherche. Pour améliorer une interface graphique et ainsi augmenter le confort visuel des utilisateurs ayant une basse vision, il faut tout d'abord savoir comment la vision fonctionne. L'œil est l'organe auquel on pense immédiatement lorsque l'on parle de vision mais il n'est rien sans le siège de la vision, le cerveau. Sans oublier tout ce qui sert à la transmission de l'information comme le nerf optique. C'est le fonctionnement en harmonie de toutes ces composantes qui donne la vision telle que nous la connaissons avec ses couleurs. Dans les différentes sections de ce chapitre, nous aborderons successivement l'oeil humain et son fonctionnement (section 2.1), l'acuité visuelle (section 2.2), la déficience visuelle (section 2.3), puis nous terminerons par les différentes propriétés des couleurs utilisées dans notre contexte (section 2.4).

Sommaire

2.1 L'œil humain et son fonctionnement **27**
2.2 L'acuité visuelle **28**
2.3 La déficience visuelle **29**
2.3.1 Troubles visuels dus à des pathologies 29
2.3.2 Regroupement des pathologies visuelles 35
2.4 Les couleurs et leurs propriétés **36**
2.4.1 Organisation des couleurs 36
2.4.2 Des propriétés diverses 37
2.5 Conclusion **41**

2.1 L'œil humain et son fonctionnement

L'œil humain est un organe complexe dont les parties sont facilement identifiables et dont leurs fonctions sont totalement connues. La combinaison des rôles joués par les différents constituants de l'œil associée à l'intervention du cerveau, forme la vision.

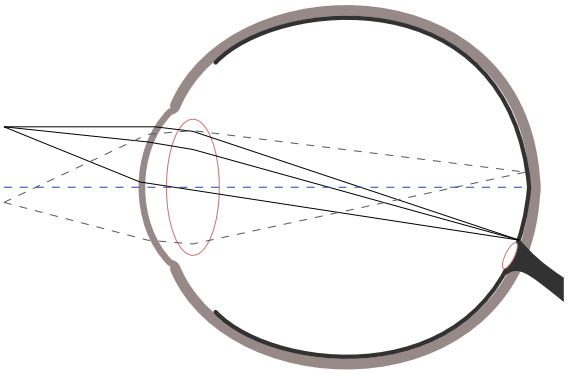


FIGURE 2.1 – L'œil humain

Le cristallin joue le rôle de lentille et permet de focaliser l'image sur la rétine. Les rayons de lumière qui arrivent à l'œil traversent le cristallin et sont orientés sur la rétine en étant déviés de leur trajectoire initiale en fonction de la distance à laquelle on regarde. Cette déviation permet d'avoir une image nette indépendamment de la distance à laquelle on regarde. Cette opération est comparable à la mise au point effectuée par le système optique d'un appareil photo. Elle est réalisée de manière automatique sans même que l'on y pense. En réalité ce fonctionnement est un peu plus complexe puisque la cornée, qui est la première chose que rencontre la lumière, joue aussi un rôle important dans la déviation des rayons lumineux. Mais il n'est pas utile pour la suite de connaître en détail ce fonctionnement.

La rétine qui reçoit ainsi l'image adaptée par le système cornée-cristallin est constituée d'une multitude de capteurs. Ces récepteurs ne sont pas tous les mêmes. Chaque groupe possède une fonction bien définie permettant l'acquisition d'informations lumineuses différentes. Les cônes sont responsables de la vision colorée, alors que les bâtonnets permettent de capter l'intensité lumineuse. Les cônes qui permettent de percevoir les couleurs se divisent en trois groupes de cellules différentes : celles qui vont être sensibles au rouge, celles qui seront sensibles au bleu et celles qui seront sensibles au vert. La combinaison de ces trois couleurs permet de créer un très grand nombre de couleurs perçues. Les cônes peuplent la partie centrale de l'œil qui nous sert à voir les détails. Les seconds types de récepteur sont les bâtonnets, ces derniers ne sont pas sensibles à la couleur mais à l'intensité lumineuse. Ils se situent principalement en périphérie de la rétine et permettent la vision dans des conditions d'éclairage faible. C'est pour cela que plus la lumière se fait rare, moins on perçoit de couleurs et tout semble devenir gris.

Le nerf optique permet le transport des influx nerveux en provenance de la rétine vers le cerveau afin que les images perçues soient traitées et analysées. Le cerveau joue un rôle primordial dans la vision puisque lors du passage dans le cristallin, l'image en provenance de l'extérieur se retrouve dessinée à l'envers sur la rétine. Le haut est en bas et la gauche passe à droite. C'est durant la phase de traitement que le cerveau retourne l'image. De plus c'est aussi grâce à lui que nous ressentons des notions de mouvement et de relief. C'est en combinant les images provenant des deux yeux qu'il est possible de créer le relief.

La couleur est la perception subjective que possède l'œil pour une ou plusieurs longueurs d'ondes lumineuses avec une amplitude donnée. La sensation de couleur, pour un

être humain, correspond à un mélange pas nécessairement équitable des trois couleurs primaires rouge, bleue et verte perçues par les cellules de la rétine (cônes). La perception d'une couleur est construite par la quantification de l'absorption maximale pour chacun des types de cônes de la rétine. L'intensité totale perçue par ces cellules sensorielles correspond à la notion de luminosité (clair ou sombre), et les intensités relatives perçues restituent la couleur. Si les couleurs vives se démarquent des autres, la limite entre ces couleurs n'est pas précise. Pour l'œil humain, il y a un continuum de couleurs dans un espace à trois dimensions, ce qui rend difficile sa représentation sur une surface comme une feuille ou un écran d'ordinateur.

L'œil humain est capable de distinguer des couleurs ayant une longueur d'onde comprise entre 400 et 800 nanomètres (environ), cette plage constitue une infime partie du spectre d'ondes électromagnétiques. En dessous des 400 nanomètres on trouve par exemple les ultraviolets et les rayons X et gamma, au dessus des 800 nanomètres il y a les infrarouges puis les ondes radios. L'œil humain pourra donc distinguer par exemple :

— les sources lumineuses, qui peuvent avoir une couleur ou non (niveau de gris).

— les couleurs pigmentaires ou chimiques car elles sont produites par la présence de colorant ou de pigment à l'intérieur de la matière observée. Ces colorants ou pigments absorbent une partie de la lumière blanche arrivant sur l'objet et de ce fait ne permettent la réfraction que d'une certaine partie du spectre lumineux. La peinture est un exemple de matière perçue comme étant de couleur grâce à des pigments.

— les couleurs structurelles ou physiques, provoquées par des phénomènes d'interférences liés à la structure microscopique de l'objet qui diffracte la lumière reçue, comme les ailes de papillon, les CDs et DVDs.

2.2 L'acuité visuelle

L'acuité visuelle est une métrique permettant d'évaluer la capacité de l'œil à discriminer deux points. En ophtalmologie il existe plusieurs manières de déterminer les facultés visuelles à partir d'échelles. Les échelles les plus utilisées sont celles de *Monoyer* qui sont des tests optométriques. D'autres échelles comme celle de *Landolt* ou celle de *Parinaud* (pour la vision de près) peuvent être utilisées. L'échelle de *Monoyer* permet d'obtenir cette valeur d'acuité visuelle en dixièmes et c'est celle qui est couramment utilisée. Pour la mesurer, on utilise des planches sur lesquelles des lignes de lettres sont écrites les unes à la suite des autres. Sur une même ligne, toutes les lettres ont la même taille. La taille des lettres change d'une ligne à l'autre. La première ligne est celle qui comporte les lettres les plus petites et la dernière celle qui a les lettres les plus grosses. Deux variantes existent, ces planches sont utilisées soit à 3 mètres soit à 5 mètres, sachant que l'on n'utilise pas les mêmes planches en fonction de la distance. La taille des lettres est proportionnelle à la distance de discrimination. On multiplie par 5 la distance de discrimination pour avoir la taille de la lettre.

La taille des lettres utilisées sur la planche dépend donc de la distance à laquelle elle est utilisée. À chaque ligne correspond une acuité visuelle donnée en dixièmes. L'acuité visuelle est définie par la formule (2.1) où VA est l'acuité visuelle, a est l'angle formé par

l'œil et les deux points à discriminer et x est le nombre de dixièmes d'acuité visuelle.

$$VA = \frac{1}{a} = \frac{x}{10} \tag{2.1}$$

Par exemple, pour une acuité visuelle de 10/10, a sera égal à $1'$ soit $0,0166°$. En utilisant les règles de base de la trigonométrie, on obtient pour une distance de 5 mètres une distance de discrimination d'environ 1.45 mm (cf. Équations (2.2)) avec d la distance de discrimination (entre deux points), D la distance entre l'œil et la planche et a l'angle de discrimination en degré.

$$\begin{aligned} d &= D \times tan(a) \\ &= 5 \times tan(0,0166) \\ &= 1.45 \end{aligned} \tag{2.2}$$

Dans ce cas, lorsque l'acuité visuelle est de 10/10 la taille de la lettre correspondante est de 7.25 mm (5 fois la distance de discrimination 1.45 mm). De la même manière, pour une acuité visuelle de 8/10 on aura (toujours à 5 mètres) des lettres de 8.75 mm, et pour une acuité visuelle de 2/10, des lettres de 35 mm.

2.3 La déficience visuelle

La déficience visuelle peut être le résultat d'une altération physique d'une partie ou de l'intégralité de l'œil. Elle peut également provenir d'un aspect psychologique. L'altération de l'œil peut être présente à la naissance, on parle alors de déficience congénitale, ou bien survenir à la suite d'un accident, d'une maladie ou d'un défaut de croissance.

2.3.1 Troubles visuels dus à des pathologies

L'œil est un organe à la fois complexe et fragile. Comme tout organe, il peut être sujet à des malformations ou des maladies. Un certain nombre d'atteintes sont considérées comme bénignes et peuvent être soit corrigées, telle l'amétropie, soit soignées. À l'inverse, il existe des atteintes de la vision dues à des pathologies aux caractéristiques très différentes, qui ne possèdent actuellement aucun traitement. Une partie de ces pathologies sont qualifiées d'évolutives car elles endommagent progressivement le champ visuel. D'autres sont héréditaires, c'est-à-dire qu'elle se transmettent de génération en génération, ou même congénitales (présentes à la naissance mais ne sont pas nécessairement héréditaires). Les termes scientifiques suivants qualifient des groupes de pathologies et nous donnent un premier aperçu de l'immense diversité des atteintes entraînant des déficiences visuelles.

L'amétropie est le terme général qui désigne l'ensemble des troubles de réfraction oculaire. Ceux-ci peuvent être partiellement ou totalement corrigés par de l'optique comme des lentilles. Les troubles désignés par l'amétropie sont relativement répandus et connus :

1. La myopie
   Elle est souvent due à une distance entre la cornée et la rétine trop importante. Cela a pour effet de former l'image en avant de la rétine. Une personne atteinte de myopie sera en difficulté seulement en vision de loin et non pas en vision de près [OMSc].
   La vision de loin est trouble, floue et nécessite un effort de concentration pour mieux percevoir les objets. La myopie peut être corrigée par des lentilles (verres, lentilles de contact).

2. L'hypermétropie
   A l'inverse de la myopie, dans le cas d'une hypermétropie, la distance entre la cornée et la rétine est insuffisante, l'image se forme par conséquent en arrière de la rétine. La vision de loin est donc correcte et c'est la vision de près qui est affectée [OMSc]. Les symptômes sont identiques à ceux de la myopie mais appliqués à la vision de près.

3. L'astigmatisme
   Il est dû à une mauvaise courbure de la cornée, celle-ci est de forme ovale au lieu d'être ronde. Dans ce cas pathologique, la vision de loin comme la vision de près en sont affectées.
   Cela a pour conséquence de rendre imprécise (floue, petits détails non visibles) la vision de loin ainsi que celle de près.

4. La presbytie
   Il s'agit d'une altération du fonctionnement du cristallin, la déformation de ce dernier est limitée et amène à une formation de l'image en avant ou en arrière de la rétine selon le contexte (comme un appareil photo dont l'autofocus aurait un dysfonctionnement et n'arriverait pas à faire la mise au point correctement) [OMSc].
   La vision de loin comme celle de près en sera affectée, il s'agit d'un manque de netteté, d'une image imprécise.

La basse vision est également un terme général désignant un groupe d'atteintes. C'est ici que l'on retrouve les déficiences visuelles provoquées par des pathologies héréditaires, congénitales, dégénératives, etc. Ces pathologies entraînent des dégradations conséquentes sur la vision. Ces dommages importants ont pour conséquence un reste visuel utilisable faible mais bien présent. La basse vision se situe donc juste avant la cécité.

1. **La DMLA**
   La Dégénérescence Maculaire Liée à l'âge (DMLA) survient généralement sur un œil sain. Elle apparaît après l'âge de 50 ans et provoque des lésions sur la rétine maculaire (le centre de la rétine, on parle parfois de la macula). C'est-à-dire qu'elle attaque la partie responsable de la vision centrale [OMSb]. Cette pathologie évolutive a des conséquences non négligeables notamment sur la lecture puisque la vision centrale est fortement sollicitée lors de cette activité. Même si des difficultés existent pour réaliser d'autres actions, la vision périphérique reste utile et permet de compenser en grande partie la perte de vision centrale.

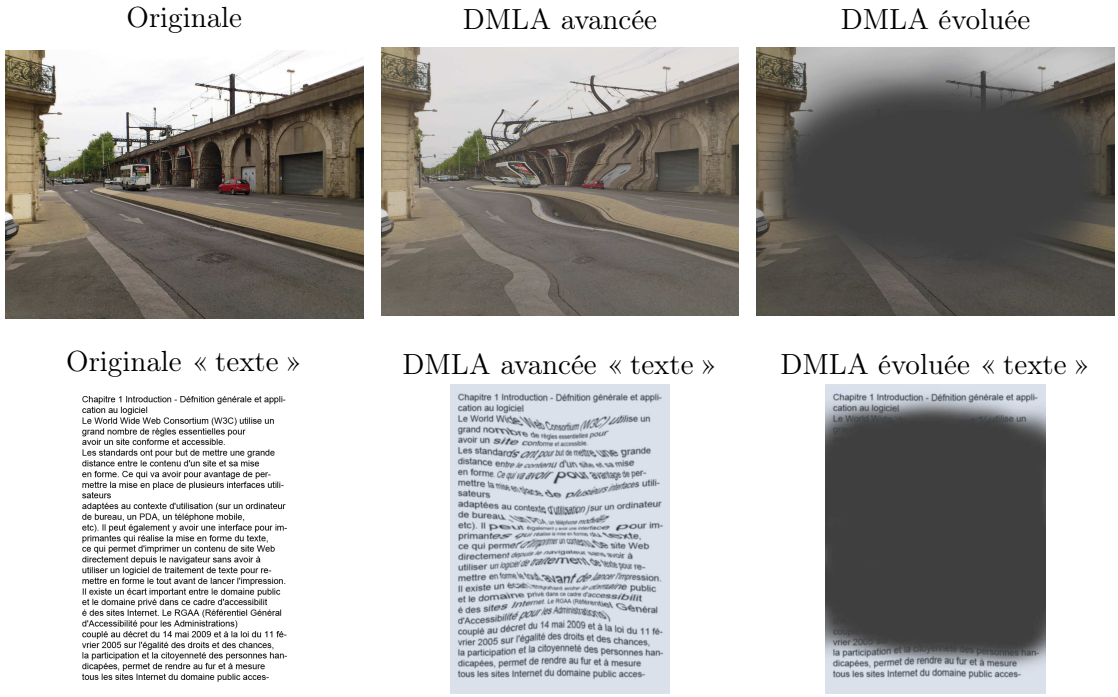


FIGURE 2.2 – Impact de la DMLA

La DMLA induit tout d'abord une déformation de la vision centrale (déformation, ondulation), par la suite, des troubles dans la perception des contrastes et des couleurs peuvent arriver, comme on peut le voir dans la figure 2.2. Il s'agit d'une pathologie évolutive.

2. **La cataracte**
   La cataracte arrive le plus souvent avec l'âge mais elle peut parfois se produire chez l'enfant : dans ce dernier cas, elle est congénitale. Sans être forcément congénitale, des cataractes précoces peuvent aussi survenir relativement tôt chez l'adulte. Il s'agit d'une opacification du cristallin qui entraîne une diminution de la quantité de lumière qui atteint la rétine [OMSa].

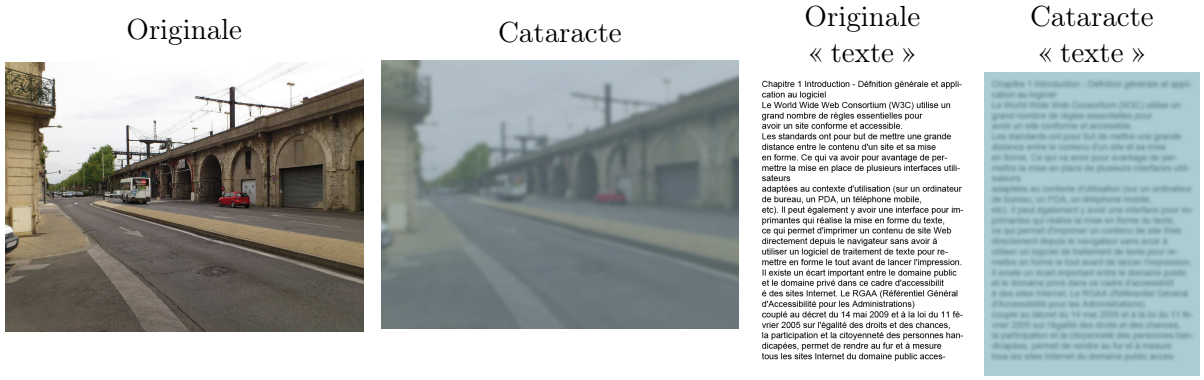


FIGURE 2.3 – Impact de la cataracte

La cataracte entraîne une baisse progressive de la vision. Cette baisse est dans un premier temps surtout constatée pour des distances importantes. En évoluant,

elle finie par être visible en permanence. Elle provoque également une diminution des contrastes (lumière et couleurs), donne l'impression d'un voile devant les yeux, comme on peut le voir dans la figure 2.3. Elle génère aussi une augmentation de la sensibilité à la lumière et facilite par conséquent les éblouissements. Tandis que la vision dans les milieux sombres est rendue plus difficile.

3. **La rétinopathie diabétique**
   Elle regroupe un ensemble de lésions de la rétine, caractéristique des personnes ayant un diabète sucre depuis plusieurs années. Dans les cas de rétinopathies diabétiques, l'évolution est prédictible. Il y a en premier lieu apparition d'occlusions et de dilatations des vaisseaux sanguins de la rétine, puis, l'évolution continue avec une rétinopathie proliférative avec l'apparition de néo-vaisseaux. L'œdème qui peut apparaître dans les phases finales diminue considérablement la vision [OMSe].



FIGURE 2.4 – Impact de la rétinopathie diabétique

La rétinopathie diabétique produit une perte de la vision des détails (principalement la vision centrale) et engendre la présence de taches mobiles sur la rétine dues à des rejets des vaisseaux sanguins. La taille des taches est variable et la gêne provoquée est notamment liée à leur emplacement et leurs déplacements. La figure 2.4 illustre les effets de cette pathologie.

4. **Le glaucome**
   Le glaucome est une perte acquise des cellules rétiniennes ganglionnaires à des niveaux bien plus importants qu'une perte « normale » (naturelle) due à l'âge. La rétine n'est pas la seule à être affectée, le nerf optique est également atteint par une atrophie. Le glaucome mène à long terme à une déficience visuelle irréversible [OMSd].



FIGURE 2.5 – Impact du glaucome

Cette pathologie entraîne une perte de la vision périphérique suivie de celle de la vision centrale comme illustré dans la figure 2.5.

5. **La rétinopathie pigmentaire**
   Il s'agit d'un terme qui regroupe un ensemble de maladies génétiques de l'œil, maladies qui affectent les photo-récepteurs dont l'œil dispose (cônes, bâtonnets) ainsi que l'épithélium pigmentaire (partie externe de la rétine). Les premiers symptômes sont une baisse de la vision nocturne puis un rétrécissement du champ visuel, la perte de la vision centrale arrive en dernier. Cette dégénérescence est due à une mutation des cellules de la rétine et elle évolue bien souvent vers la perte totale de vision (cécité) lorsque toutes les cellules sont atteintes.

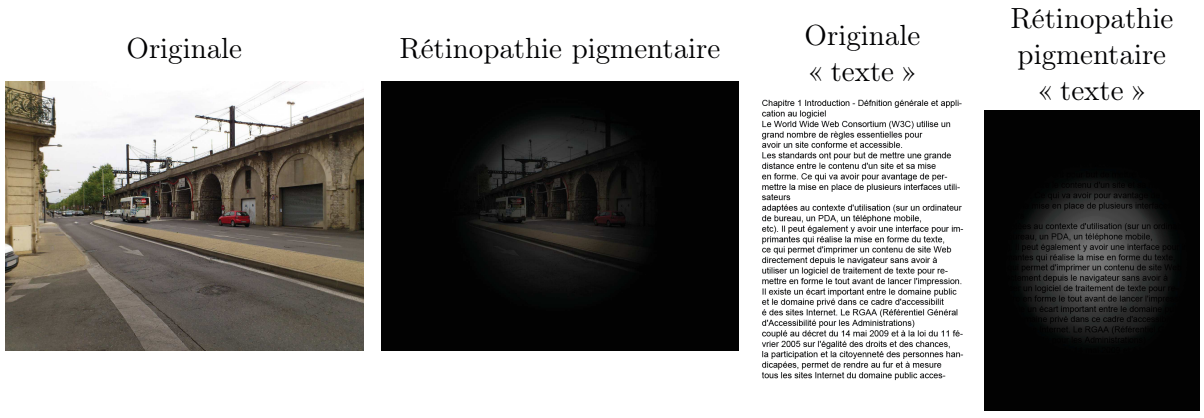


FIGURE 2.6 – Impact de la rétinite pigmentaire

La rétinopathie pigmentaire engendre une diminution progressive de la vision, une perte de la vision périphérique, une diminution des contrastes, une augmentation de la sensibilité à la lumière comme illustré dans la figure 2.6. L'adaptation au changement de lumière est difficile. Il y a une perte de la vision nocturne et donc aussi des sources de lumière très faibles. Cette baisse progressive se termine une fois la cécité atteinte.

6. **L'albinisme**
   L'albinisme est une maladie génétique héréditaire qui est à l'origine d'une faible production de mélanine voire même d'une production nulle. Elle touche donc l'ensemble du corps humain et notamment l'iris et la rétine. Cette dernière a un déficit important en récepteurs et pigments et ce déficit est d'autant plus important que l'on se rapproche de la fovéa. La fovéa est la partie centrale de la macula, elle est formée uniquement de cônes. En alignement avec l'axe optique de l'œil, c'est cette zone qui offre la meilleure résolution optique. Le nerf optique peut également présenter une hypoplasie (développement insuffisant) et la distribution des fibres nerveuses entre les deux yeux est anormale.
   La vision de près est faible mais des aides visuelles permettent une acuité relativement correcte. La vision de loin est très diminuée (en dessous de $4/10^{e}$).

7. **La névrite optique**
   Cette pathologie correspond à une inflammation du nerf optique pouvant entraîner une perte partielle ou bien totale de la vision. Cette inflammation du nerf peut se trouver en arrière de l'œil comme à la base du nerf au niveau du globe oculaire.



FIGURE 2.7 – Impact de névrite optique

Des zones d'absence de vision apparaissent dans le champ de vision, les couleurs et contrastes deviennent difficiles à distinguer, tout comme les objets lorsqu'ils se trouvent en pleine lumière comme illustré dans la figure 2.7.

8. **L'hémianopsie homonyme**
   Cette pathologie est causée par une mauvaise transmission de l'information visuelle au cerveau. Des affections du cerveau incluant les tumeurs, les maladies inflammatoires ainsi que les traumatismes peuvent déclencher une hémianopsie. Elle est le plus souvent due à un accident vasculaire cérébral (AVC).

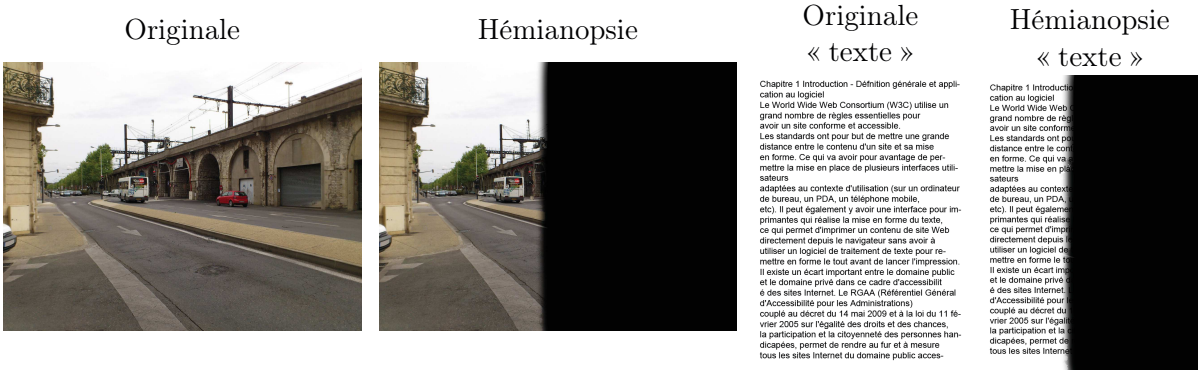


FIGURE 2.8 – Impact de l'Hémianopsie homonyme

Cette pathologie entraîne la perte de la moitié du champ visuel de la personne, et ce de manière symétrique (du même côté) pour les deux yeux (figure 2.8). Cette perte apparaît suivant un plan vertical, donc la vision sera soit altérée à gauche, soit à droite. Cela rend difficile la lecture et les personnes atteintes sont souvent surprises par des taches surgissant « de nulle part ».

Cet aperçu ne couvre que partiellement les pathologies, symptômes et atteintes existantes. De plus les vignettes représentent ici une idée générale de la vision en fonction d'un type de déficience. Dans la plupart des cas, la vision altérée ne se résume pas à ces images, chaque individu a sa propre évolution et bien souvent il s'agit de combinaisons entre plusieurs atteintes.

2.3.2 Regroupement des pathologies visuelles

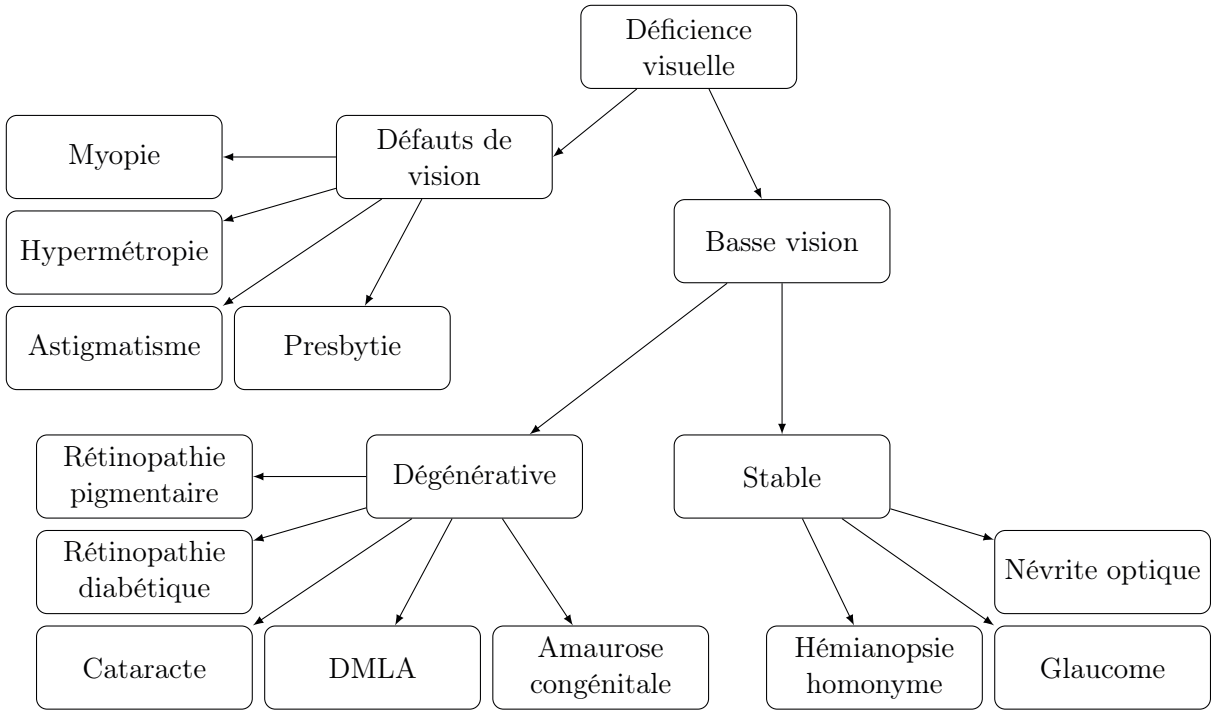


FIGURE 2.9 – Classification de l'ensemble des pathologies support

La déficience visuelle est un terme très général qui regroupe beaucoup de pathologies différentes. Ce que l'on appelle la basse vision n'est pas la seule catégorie des déficiences visuelles, l'amétropie (ou défauts de vision) en fait également partie. L'amétropie regroupe toutes les atteintes pouvant être corrigées par de l'optique (myopie, presbytie, hypermétropie etc...). Même si ces dernières sont très répandues et facilement corrigées elles n'en restent pas moins des pathologies visuelles pouvant à un certain niveau engendrer de véritables difficultés. De son côté la basse vision regroupe des pathologies « plus lourdes » qui provoquent des difficultés dans la vie quotidienne bien plus importantes. Un grand nombre d'entre elles ne possèdent pas, à l'heure actuelle, de véritables traitements. La Figure 2.9 donne un classement de plusieurs pathologies visuelles qu'elles soient corrigibles ou non.

2.4 Les couleurs et leurs propriétés

La perception des couleurs découle de la représentation mentale des stimuli reçus par le cerveau en provenance des photorécepteurs rétiniens de l'œil. Ces couleurs permettent notamment d'identifier les objets, de les distinguer les uns des autres, de leur donner une signification particulière ou même de porter des significations conventionnelles telles que le danger (rouge).

2.4.1 Organisation des couleurs

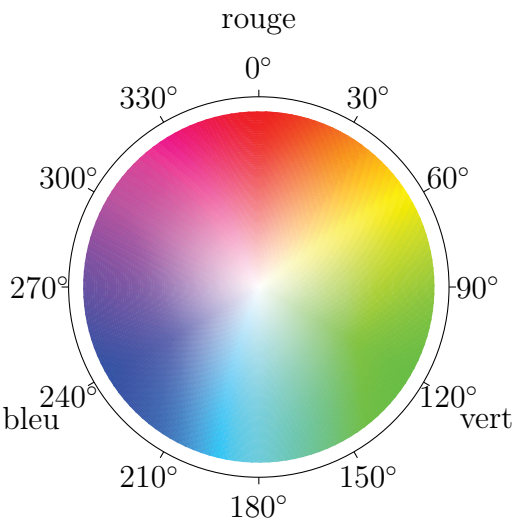


FIGURE 2.10 – Disque chromatique de saturation par synthèse additive

Différents systèmes de synthèse des couleurs sont apparus au fil du temps. Ils sont souvent liés à l'évolution des techniques de représentation (peinture, teinture, impression, écrans, etc.). Chaque système utilise un certain nombre de couleurs primaires : 3 couleurs sont primaires entre elles si aucun mélange de deux de ces couleurs ne peut produire la troisième. Les deux systèmes de synthèse les plus connus et couramment utilisés sont la synthèse additive utilisant les trois couleurs primaires Rouge, Vert, Bleu (RVB) et la synthèse soustractive utilisant les trois couleurs primaires Cyan, Magenta, Jaune (CMJ).

Les couleurs, que ce soient des couleurs primaires ou bien des couleurs secondaires obtenues par mélange de ces trois couleurs primaires, peuvent être organisées sous la forme d'un disque. Les couleurs primaires y apparaissent disposées à 120°les unes par rapport aux autres. Ce disque est souvent appelé disque ou cercle chromatique et est utilisé dans beaucoup de domaines comme la peinture, la photographie, le design, l'infographie, etc.

Plusieurs représentations du cercle chromatique existent en fonction des trois couleurs primaires de base qui sont utilisées, mais aussi en fonction de ce que l'on cherche à représenter comme la saturation ou la brillance. La figure 2.10 représente le disque chromatique pour la saturation. Le centre de ce disque est exempt de toute couleur et celles-ci s'intensifient au fur et à mesure que l'on s'éloigne du centre. Lorsque l'on atteint la circonférence du disque, on a les couleurs les plus saturées.

Concernant le « repère » de base (les trois couleurs primaires utilisées pour les mélanges), deux groupes sont très souvent utilisés. Le premier est RVB (Rouge, Vert et Bleu) et le second est CMJ (Cyan, Magenta et Jaune). Dans les domaines techniques (audiovisuel), on a l'habitude de parler et de manipuler comme couleurs primaires, le rouge, le vert et le bleu. Cependant un exemple simple de repère CMJ existe puisque dans les imprimantes photos, on trouve fréquemment du cyan, du magenta et du jaune comme couleurs de base. Ceci est également vérifié dans de nombreux processus de photocomposition.

Indépendamment de la saturation ou des trois couleurs de base utilisées, le cercle peut être soit une représentation par absorption (synthèse soustractive) soit une représentation par addition (synthèse additive). Soit on considère les couleurs de base comme étant une source de lumière alors on se trouve dans le cadre d'une synthèse additive et la somme des trois couleurs de base donne du blanc. Soit, à l'inverse si on se trouve dans le cadre d'une synthèse soustractive comme en présence de pigments de couleurs (peinture), ces derniers absorbent la lumière et ne renvoient qu'une partie du spectre. Alors le mélange des trois couleurs de base donne du noir.

Deux couleurs sont dites complémentaires l'une de l'autre si leur mélange donne lieu à une absence de chromaticité (blanc, gris, noir). Autrement dit, la résultante de ce mélange est une couleur neutre. La couleur complémentaire se détermine soit par le calcul, soit de manière graphique à partir des différents cercles/disques chromatiques comme celui décrit ci-dessus. Il s'agit dans ce dernier cas, pour une couleur donnée, de la couleur qui lui est diamétralement opposée. Ainsi, le jaune est le complémentaire du bleu, le magenta est le complémentaire du vert et le cyan celui du rouge. Les paires de couleurs complémentaires possèdent certaines propriétés notamment de contrastes.

Une couleur pure est une couleur qui n'a subi aucun ajout de noir ou de blanc. Plus formellement, c'est une couleur qui a la plus grande saturation parmi les couleurs d'une même teinte. Dans le cas du disque chromatique décrit précédemment qui représente le niveau de saturation des couleurs, les couleurs pures se trouvent par conséquent à la périphérie (circonférence) de ce dernier. La circonférence du disque représente par conséquent l'ensemble des couleurs pures possibles.

2.4.2 Des propriétés diverses

La couleur est à l'origine de plusieurs propriétés ou grandeurs physique qui permettent de la qualifier. Certaines propriétés correspondent à une seule et unique couleur tandis que d'autres apparaissent lorsque l'on met en relation plusieurs couleurs.

La luminance (ou brillance) relative

La luminance d'une couleur est une grandeur numérique permettant de qualifier son intensité lumineuse, c'est-à-dire la quantité de lumière émise par la couleur. Pour calculer cette grandeur, on se reporte aux intensités lumineuses de chaque composante de la couleur (composantes primaires). Il existe plusieurs façons de mesurer une luminance. Le modèle de représentation TSL (Teinte Saturation Luminance) utilise une luminance relative (pourcentage de luminance). Elle a l'avantage d'être facilement calculable, mais l'inconvénient majeur est qu'elle ne correspond pas vraiment à ce que l'on voit. Dans le but de combler ce défaut, le W3C a publié une manière de calculer ce que l'on appelle la luminance ou brillance perçue. Cette dernière tient compte du fait que nous ne voyons pas toutes les couleurs de la même manière. En effet le vert nous paraîtra bien plus lumineux que le rouge qui lui-même paraîtra plus lumineux que le bleu. Des coefficients ont donc été définis afin d'évaluer cette différence. Les coefficients donnés peuvent varier. Cependant la relation d'ordre « vert plus lumineux que rouge plus lumineux que bleu » reste la même. Le W3C fournit un coefficient pour chaque composante et une formule pour déterminer la brillance d'une couleur en particulier. Cette intensité lumineuse calculée est la base du calcul des contrastes de luminance. La formule 2.3 donne la luminance relative L pour le modèle de représentation RGB.

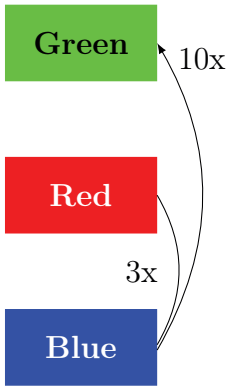


FIGURE 2.11 – Différence de brillance pour les trois couleurs primaires R, v et B

$$L = 0.2126 \times R + 0.7152 \times G + 0.0722 \times B \tag{2.3}$$

Dans la formule 2.3, R, G et B sont les valeurs normalisées de RsRGB, GsRGB et BsRGB. Lors de la normalisation de chaque composante de la couleur, on tient compte de l'intensité de cette composante. La perception de l'intensité lumineuse d'une couleur n'est pas la même selon que la couleur soit vive ou non. Une condition, permet selon le cas, d'appliquer des calculs différents. Pour comprendre les formules suivantes, qui indiquent respectivement comment calculer les valeurs normalisées de R, G et B, il faut comprendre que 0.03928 est la valeur seuil pour déterminer si une couleur est vive.

— **Si** $RsRGB \leq 0.03928$ **alors** $R = RsRGB/12.92$ **sinon** $R = ((RsRGB{+}0.055)/1.055)^{2.4}$
— **Si** $GsRGB \leq 0.03928$ **alors** $G = GsRGB/12.92$ **sinon** $G = ((GsRGB{+}0.055)/1.055)^{2.4}$
— **Si** $BsRGB \leq 0.03928$ **alors** $B = BsRGB/12.92$ **sinon** $B = ((BsRGB{+}0.055)/1.055)^{2.4}$

Avec RsRGB, GsRGB, et BsRGB définis comme suit :

— $RsRGB = R8bit/255$
— $GsRGB = G8bit/255$
— $BsRGB = B8bit/255$

Le modèle de représentation sRGB est simplement le modèle standard de RGB pour le Web (RVB en français). Il se base sur les mêmes 3 composantes rouge, verte et bleue primaires. Chacune de ces trois composantes est codée sur 8 bits (1 octet) et possède donc une valeur comprise entre 0 et 255. Pour effectuer le calcul de la luminance relative, chaque valeur des composants de base est donc ainsi ramenée entre 0 et 1.

Le contraste de luminance

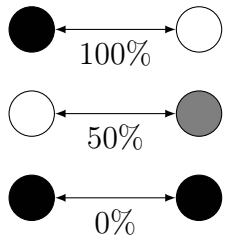


FIGURE 2.12 – Exemples de contrastes de brillance

Le terme de contraste désigne une différence entre deux choses. Lorsqu'il s'agit de couleurs il existe plusieurs types de relations de contraste. Actuellement 7 relations de contraste sont identifiées mais certaines d'entre elles sont très peu connues et donc pas vraiment utilisées. Les principales sont le contraste de chrominance et le contraste de luminance. Le contraste de chrominance est difficile d'utilisation car il dépend fortement de la façon dont les individus perçoivent les couleurs. A l'inverse le contraste de luminance est bien défini et se calcule très simplement. Le contraste de luminance est celui qui est largement utilisé dans les normes et standards et c'est aussi celui-ci que nous utiliserons par la suite. On peut s'imaginer le contraste de luminance comme si on passait les deux couleurs en niveau de gris et que l'on regardait à quel point elles sont différentes. Le contraste de luminance entre deux couleurs permet de déterminer la distance qui les séparent en terme d'intensité lumineuse perçue. Il s'agit d'une grandeur numérique correspondant à la distance qui existe sur chaque composante primaire des deux couleurs. Le W3C a publié une échelle et une formule permettant de calculer le contraste de luminance entre deux couleurs en fonction de la perception lumineuse de chaque composante primaire. Ce calcul est fourni dans la formule 2.4 suivante où le terme en 0.05 sert à arrondir :

$$Contrast = \frac{L1 + 0.05}{L2 + 0.05} \tag{2.4}$$

Avec :

$L1$ et $L2$ la luminance relative pour la première et la seconde couleur

La valeur obtenue par cette formule est une valeur entre 1 et 21, souvent écrite de 1:1 (1 pour 1) à 21:1 (21 pour 1). Un contraste de 1 (1:1) est un contraste nul, c'est-à-dire que les deux couleurs ont strictement la même luminance. Lorsque le contraste atteint 21, la différence de luminance est maximale. Cette valeur pour le contraste de luminance permet notamment d'évaluer les règles sur le contraste minimal dans le Web Content Accessibility Guidelines (WCAG 2.0). Par la suite, cette valeur sera parfois reportée sous la forme d'un pourcentage de manière à avoir une valeur plus intuitive et compréhensible.

Les contrastes simultanés

Le contraste simultané désigne un phénomène naturel (chez l'homme) qui a pour conséquence, pour une couleur donnée, que l'œil exige en quelque sorte son complémentaire, et ce au même instant (simultanément). Par conséquent, si le complémentaire de la couleur n'est pas physiquement présent là où l'œil s'attend à le trouver, il le génère « virtuellement ». De manière générale, plus la durée de fixation de la couleur prédominante est prolongée mais aussi plus celle-ci est intense (lumineuse), plus le phénomène de contraste simultané est marqué.

L'effet de contraste simultané se produit entre une couleur et un gris mais aussi entre deux couleurs pures qui ne sont pas exactement complémentaires l'une de l'autre. Chacune des deux couleurs cherche à repousser l'autre, vers sa complémentaire. Les couleurs paraissent alors dans un état d'excitation dynamique. Leur stabilité disparaît et elles vibrent.

Il est possible d'atténuer ce phénomène de deux façons différentes selon que ce dernier se produit entre deux couleurs ou bien entre une couleur et un gris. Dans le cas où il se produit entre une couleur et un gris, il est possible de l'atténuer en teintant légèrement le gris avec la couleur qui lui est associée. Dans l'autre cas il suffit d'augmenter la distance d'une des deux couleurs avec le complémentaire de l'autre selon le cercle chromatique.

Le contraste simultané est un phénomène connu et maîtrisé. Il peut être mesuré et ce de manière assez précise [wik15]. Il est donc possible de créer volontairement ce phénomène et de l'accentuer ou, au contraire, dans le cadre de notre travail, de le réduire pour améliorer la facilité de lecture.

Voici un exemple concret montrant le phénomène de contraste simultané. Deux images colorées distinctes possèdent chacune des rayures grises **identiques**. Sur la vignette de gauche, la couleur rouge ayant pour complémentaire le bleu, les rayures grises paraissent légèrement bleuâtres. Tandis que sur la vignette de droite, la couleur bleue ayant pour complémentaire le rouge, les bandes grises paraissent rosâtres.



FIGURE 2.13 – Contrastes simultanés

Même si une page ne génère pas ce phénomène par le jeu de couleurs utilisé, l'altération de cette page pour répondre à la contrainte de contraste peut entraîner ce dernier. D'où l'importance d'une méthode de calcul du contraste simultané.

Taille perçue selon la brillance

La brillance de la couleur d'une image a un impact sur la manière de percevoir les objets notamment au niveau de leur dimension. Du fait de certaines propriétés de la lumière comme la réfraction, des objets réfracteurs ne seront pas perçus de la même manière que des objets émetteurs de lumière.



FIGURE 2.14 – Perception des dimensions selon la couleur

Sur la vignette de gauche, le carré central (blanc) apparaît plus grand que le carré central (noir) de la vignette de droite, et pourtant, ils sont exactement de la même taille. L'œil ne percevant que la lumière, le noir correspondant à une absence de lumière, et étant donné que les taches lumineuses ont tendance à affecter légèrement les récepteurs proches de ceux concernés sur la rétine, la zone blanche s'étale sur la zone noire donnant cette impression de taille différente. D'où le fait de dire que « le noir amincit ». Ce phénomène ne se produit pas qu'avec des formes géométriques, il se produit aussi pour du texte. Par conséquent un texte noir sur un fond blanc (affichage le plus répandu) paraîtra plus petit que le même texte en blanc sur un fond noir. Cela peut donc jouer sur la lisibilité du texte en plus du fait qu'un fond blanc lumineux peut générer une gêne, voire même un éblouissement.

2.5 Conclusion

Dans ce chapitre, nous avons présenté différentes notions sur l'oeil, l'acuité visuelle, les déficiences visuelles et décrit plusieurs propriétés sur les couleurs, avec des formules permettant notamment d'évaluer la luminance et le contraste de luminance. Ces deux notions seront utilisées pour adapter des éléments sur les pages Web, comme par exemple un texte sur son arrière-plan. Les contrastes simultanés seront au contraire des phénomènes à éviter lorsque les couleurs seront modifiées sur une page. Dans le chapitre suivant, nous développons un état de l'art sur l'accessibilité du Web, domaine dans lequel se situe notre contribution.

Chapitre 3

État de l'art

Préambule

Ce chapitre se consacre à l'étude des technologies, outils et travaux de recherche existant autour de l'accessibilité au Web pour les personnes ayant une basse vision. Nous commençons par décrire les fondamentaux de ce qu'est réellement une page Web et les différents standards et normalisations qui existent et permettent d'assurer une accessibilité minimale. Nous détaillons ensuite les diverses possibilités offertes par les technologies d'assistance les plus connues et les outils spécifiquement dédiés à certaines utilisations. Certaines limites de ces outils sont également abordées succinctement. Les divers travaux de recherche existant dans le domaine sont ensuite étudiés et comparés. Ce chapitre est complété par le chapitre suivant qui explicite de manière concrète les limites des divers outils d'assistance sur des exemples mettant en avant les difficultés pouvant être rencontrées.

Sommaire

3.1 Le Web dans sa forme visuelle **44**
3.2 La normalisation **45**
3.2.1 Domaine d'application et cadre juridique 46
3.2.2 Les éléments sur lesquels portent les normes et recommandations 46
3.2.3 Le contenu Web 46
3.2.4 Les outils de production 47
3.2.5 Les outils de consultation 48
3.2.6 Un exemple plus concret 48
3.2.7 Les limites 49
3.3 Les technologies d'assistance **49**
3.3.1 Les *magnifiers* d'écran 49
3.3.2 Intégration dans les systèmes d'exploitation 53
3.3.3 Les Outils de gestion des styles 54
3.4 Adaptation de pages Web et préférences utilisateur **56**
3.4.1 Architecture de transformation, destinataires et but 56
3.4.2 Éléments traités, reconnaissance automatique, cible d'application, partie transformée et conservation du contexte originel . . 57

3.4.3 Profil utilisateur : langage de définition, acquisition, déficience spécifique visée et définition a priori 59
3.4.4 Méthode d'adaptation 60
3.4.5 Évaluation, métriques et utilisateurs 60
3.4.6 Démonstrateur et corpus de données fourni 61
3.5 Conclusion **61**

3.1 Le Web dans sa forme visuelle

On appelle interface graphique la partie visuelle d'un site, application ou service en ligne. C'est cette interface qui permet à l'utilisateur d'acquérir des informations mais aussi d'interagir avec elles et l'on parle alors d'Interaction Homme Machine. Ces interactions se déroulent habituellement par le biais d'un périphérique d'entrée comme la souris, le clavier ou un écran tactile. Ce sont des périphériques physiques et préhensibles mais le contrôle (retour), lui, est visuel. Dans le cas de l'interaction avec une page Web, il est également possible d'utiliser un lecteur d'écran avec une synthèse vocale ou un retour en braille.

L'interface graphique est constituée de plusieurs parties, il y a d'un côté ce que l'on peut appeler le contenu comme le texte, les images, les formulaires, les boutons ou les menus et d'un autre côté la charte graphique. Cette charte définit une architecture de page commune pour tout ou partie du site Web. Elle permet de choisir la palette de couleurs, de définir la taille et la position de certains éléments comme le menu ou la barre de titre. Elle définit également le logo ainsi que tous les autres détails du même ordre. La conception de la charte graphique se fait généralement séparément de la conception technique du site Web. Son intégration dans le reste du site est un travail à part entière dans la phase de développement.

Dans le cadre d'applications ou de services, l'interface graphique dispose d'éléments tels que des boutons d'action, des champs de saisie, des zones de visualisation ou de manipulation. À la différence des sites Web de contenu, qui sont statiques du point de vue de l'utilisateur, les applications Web ne nécessitent pas forcément un rechargement de la page à chaque action. La succession des pages et l'évolution des contenus peut être générée sur la machine de l'utilisateur. Seules des données sont transférées pour faire des calculs, des statistiques ou rafraîchir une portion de l'écran. Cette utilisation dynamique côté client (utilisateur) peut être réalisée à l'aide de langages de programmation exécutés sur la machine de l'utilisateur comme Javascript. Cette technologie permet de modifier des éléments en termes de position, de taille, de couleur, de nom, d'animation, de temps d'affichage, ou même d'actions. En résumé, elle permet de modifier les caractéristiques propres d'un objet afin de le transformer et de l'adapter à une nouvelle situation.

La mise en place de la charte graphique passe majoritairement par ce que l'on appelle des feuilles de style. La technologie de feuilles de styles en cascade (CSS) permet d'appliquer, avec possibilité de surcharge, des styles et des règles d'affichage. Elles permettent d'appliquer un style pour chaque élément ou groupe d'éléments correspondant à un certain critère à la manière des expressions régulières. Ainsi un style pour les titres de premier niveau pourra être défini, évitant de le faire pour tous les titres un à un. Il est également possible de donner un identifiant à un élément HTML afin de le cibler dans la feuille de style pour appliquer un style ne se rapportant qu'à cet objet. Elles permettent de sélectionner, cibler des éléments et des sous-éléments suivant des règles et une navigation. Par exemple, on pourra ne cibler que les liens se trouvant dans le bloc HTML identifié par un label « menu ».

La conception de sites et d'applications Web ne se fait pas complètement au hasard, elle se doit dans certains cas de répondre à des normes et de se plier à la législation.

Les enjeux de l'accessibilité du Web sont importants. Internet est un outil à grand

potentiel en pleine expansion et est de plus en plus présent dans la vie quotidienne de tous. Son grand potentiel réside notamment dans l'accès à une importante quantité d'information de toutes sortes et dans l'interaction avec divers services. Pour les personnes à mobilité réduite (PMR), il permet un accès à la culture, à l'information et aux services. Internet apporte par conséquent des prestations plus ou moins identiques à celles habituelles de la vie quotidienne comme par exemple la consultation de comptes bancaire en ligne, les fils d'informations, l'échange de courrier, ou bien le commerce en ligne.

L'accessibilité Web et plus généralement l'accessibilité numérique est un enjeu majeur pour l'égalité des droits et des chances face aux TIC (Technologies de l'information et de la Communication). En particulier, elle vise à faciliter ou permettre l'utilisation des logiciels, la navigation ainsi que l'interaction sur les sites Web. L'accessibilité Web concerne les personnes en situation de handicap quelles qu'elles soient, mais elle répond également aux besoins des utilisateurs seniors ou des personnes se trouvant dans un contexte environnemental contraignant. Dans le cadre d'un développement Web, il est montré que lorsqu'un site est développé en respect des règles et recommandations d'accessibilité, il dispose d'un avantage non négligeable en terme de qualité d'information et d'organisation de celle-ci, de référencement (par exemple sur les moteurs de recherche), d'acquisition des contenus et d'utilisabilité.

Les enjeux de l'accessibilité numérique sont nombreux (politiques, économiques, sociaux ...) et sont de plus en plus présents. L'accessibilité numérique tend en fin de compte à servir toute personne qu'elle soit valide ou non.

3.2 La normalisation

Sur Internet, les ressources de tous types et les différents supports de communication doivent dans certains cas respecter des normes/standards. C'est notamment sur leurs interfaces, qui permettent aux utilisateurs d'accéder à l'information ou d'utiliser un service, que ces normes ou réglementations s'appliquent. Les pathologies et symptômes et par conséquent les besoins des utilisateurs étant très différents et même régulièrement contradictoires d'une personne à l'autre (d'un handicap à l'autre), les normes se doivent par conséquent d'être généralistes et ouvertes. On cherchera donc par cette réglementation à rendre les interfaces accessibles au plus grand nombre tout en permettant à des technologies d'assistance tierces de compléter l'accessibilité pour des besoins plus particuliers.

La législation Française, par la loi *2005-102* sur l'égalité des droits et des chances et la citoyenneté des personne handicapées, ainsi que le décret *2009-543* mettant en application la loi *2005-102*, définissent un cadre juridique et une série de normes encadrant le développement. Ces normes constituent le Référentiel Général d'Accessibilité pour les Administrations (RGAA [Dir09]). Ce référentiel évolue au fil du temps et de nouvelles versions sortent améliorant les précédentes. Actuellement nous en sommes à la troisième version de ce référentiel. Le *World Wide Web Consortium* (W3C) est à l'origine de la standardisation des langages HTML et CSS. Il a été fondé en octobre 1991 par Tim Berners-Lee (l'inventeur du World Wide Web (www)). Il s'agit d'un consortium international co-dirigé par plusieurs organisations. L'objectif premier du W3C est de créer des standards pour le Web et de publier des recommandations. Il travaille entre autres dans le domaine de l'ac-

cessibilité par le biais d'initiatives comme WAI-ARIA [Web] (Web Accessibility Initiative - Accessible Rich Internet Applications).

3.2.1 Domaine d'application et cadre juridique

Légalement tous les sites, applications ou services Web ne sont pas soumis à une obligation d'accessibilité. Les développeurs de sites personnels (particuliers) ne sont pas concernés par cette obligation mais peuvent apporter un plus non négligeable à leur site en s'y intéressant.

Le RGAA introduit par le décret *2009-546* et la loi *2005-102* concerne l'ensemble des ressources numériques du domaine public et des services de première nécessité. Ce référentiel encadre une obligation pour ces services de rendre accessible leurs sites, applications, services et ressources. Ils disposent donc d'une échéance pour atteindre cette accessibilité sous peine de sanction.

AccessiWeb est une marque déposée qui appartient au W3C par le biais du WAI-ARIA. Son objectif est de rendre opérationnels les normes et standards définis par le W3C. Le WCAG 2.0 [WCA] définit les normes et standards relatifs aux contenus Web. D'autres ensembles de normes pour les outils de consultation ou de production existent. Ces normes peuvent s'appliquer à n'importe quel site ou outil et ne se limitent pas aux administrations (RGAA). Toute personne désireuse de rendre son site Web ou ses outils de consultation et de production accessibles peut se baser sur cette référence afin d'orienter son développement.

3.2.2 Les éléments sur lesquels portent les normes et recommandations

L'accessibilité du Web repose sur plusieurs domaines et étapes de la conception et du développement d'un site ou service Web. Ces différentes étapes et domaines sont interdépendants. Le contenu Web constitue l'ensemble de l'information (textuelle ou non) qui est transmise par la structuration du code (à l'aide de balises). Les outils d'évaluation représentent l'ensemble des outils permettant une analyse du contenu pour déterminer la conformité aux normes d'accessibilité. Les outils d'édition de contenu regroupent tous les outils qui permettent de générer de manière automatique du contenu Web. Les agents sont l'ensemble des applications qui permettent le rendu final du contenu à l'utilisateur (typiquement les navigateurs). Et enfin le dernier domaine est celui de l'humain avec les développeurs de contenu Web, d'outils d'édition ou d'évaluation et d'agents.

Les standards donnés par le WAI-ARIA se décomposent en trois grand axes principaux : les contenus Web, les outils de consultation et les outils de production de contenu.

3.2.3 Le contenu Web

L'ensemble des fournisseurs de contenu Web est concerné par les recommandations sur le contenu. Ces directives sont les « Web Content Accessibility Guidelines » (WCAG

[WCA]). Les WCAG 1.0 sont sorties en 1999 et ont permis de donner un grand nombre de recommandations et normes sur le contenu. Celles-ci ont par la suite évolué jusqu'à donner les WCAG 2.0 qui sont la référence depuis 2008.

Alors que les WCAG 1.0 portent principalement sur le langage HTML, les WCAG 2.0 ont pour objectif d'être indépendantes du langage et par conséquent de s'adapter à tous les langages existants et futurs. On trouvera par exemple comme grandes lignes :

— Des contenus perceptibles (fournir des alternatives textuelles à tous les éléments non textuels, améliorer la lecture des éléments textuels)

— Des contenus utilisables (rendre utilisables toutes les fonctionnalités au clavier, garantir un temps suffisant pour la compréhension et l'utilisation d'un contenu)

— Des contenus compréhensibles (texte lisible et compréhensible, comportement prévisible)

— Des contenus robustes (optimisation de la compatibilité entre les agents utilisateurs)

Chacun de ces points se divise en plusieurs « critères de succès » correspondant à un niveau A, AA ou AAA. Ces critères constituent la base de l'évaluation de l'accessibilité d'un contenu Web.

3.2.4 Les outils de production

En ce qui concerne les applications de production de code source HTML (éditeur de pages) ou les services de publication en ligne, la référence est les « Authoring Tools Accessibility Guidelines ». Les ATAG 1.0 sont en cours d'évolution pour donner les ATAG 2.0 [ATA], une version adaptée aux systèmes apparus depuis la première version, comme les systèmes de gestion de contenu, les blogs... Ces recommandations visent d'une part à avoir un outil de génération de code, de publication ou de gestion de page accessible à tous, et d'autre part de prévoir l'accessibilité du contenu créé par le biais de ces outils. Les points suivants font donc partie des normes :

— L'accessibilité de l'interface utilisateur par :
  — La compatibilité avec les différentes technologies d'assistance.
  — La capacité à être perceptible, compréhensible et utilisable.

— La capacité des outils à inciter à la génération de contenus accessibles :
  — Guider les utilisateurs dans l'édition de contenus accessibles par le biais de l'outil.
  — Inciter les utilisateurs à mettre en œuvre l'accessibilité des contenus.

3.2.5 Les outils de consultation

Pour que les contenus accessibles puissent être exploités, il est nécessaire d'avoir des outils de consultation de contenu Web, qui respectent eux aussi des règles. Les « User Agent Accessibility GuideLines » (UAAG [UAA]) permettent de règlementer cela en proposant des directives. Les UAAG 2.0, tout comme les ATAG 2.0, sont en cours d'élaboration pour s'adapter aux besoins actuels et aux technologies futures.

Les UAAG 2.0 sont décomposées en 5 principes généraux qui eux-même sont composés de directives.

— respecter les normes et conventions applicables ;
— favoriser l'accès par les technologies d'aide ;
— garantir que l'interface utilisateur soit perceptible ;
— garantir que l'interface utilisateur soit utilisable ;
— garantir que l'interface utilisateur soit compréhensible.

3.2.6 Un exemple plus concret

Dans le cas d'un contenu Web, de manière plus concrète, les standards et recommandations vont notamment porter sur le contraste des éléments ou la lisibilité.

AccessiWeb donne par exemple comme recommandation pour les couleurs : « Ne pas donner l'information uniquement par la couleur et utiliser des contrastes de couleurs suffisamment élevés. » On trouve une autre recommandation pour les tableaux : « Donner à chaque tableau de données, un résumé et un titre pertinent, identifier clairement les cellules d'en-tête, utiliser un mécanisme pertinent pour lier les cellules de données aux cellules d'en-tête. Pour chaque tableau de mise en forme, veiller à sa bonne linéarisation ». Pour les images on trouvera : « Donner à chaque image porteuse d'information une alternative textuelle pertinente et une description détaillée si nécessaire. Remplacer les images textes par du texte muni d'un style lorsque c'est possible ».

En plus de ces trois exemples on trouve des recommandations sur :

— Les liens.
— Les cadres.
— Le multimédia.
— Les éléments obligatoires.
— La présentation de l'information.
— La navigation.
— Les scripts.
— La structuration de l'information.
— La formulation.
— La consultation.

Au-delà de ces grandes lignes, il y a pour chaque recommandation des éléments plus précis [W3C15]. Pour le contraste on trouvera par exemple une valeur minimale demandée de 4.5:1 pour l'ensemble des éléments en dehors de quelques exceptions détaillées. La valeur peut varier selon la taille des éléments à percevoir. Ces valeurs sont données sur une échelle de contraste allant de 1 à 21 (1:1 à 21:1) dont la méthode de calcul est définie par le W3C.

3.2.7 Les limites

Ces recommandations ont été définies suite à l'étude des besoins des personnes déficientes visuelles. Ces normes sont catégorisées et nivelées ce qui permet d'avoir une validation des interfaces Web plus ou moins rigoureuse (A, AA ou AAA). Le niveau qui cherche le plus souvent à être atteint par l'ensemble des interfaces est le niveau intermédiaire (AA).

Les normes actuelles tendent à rendre accessibles les interfaces Web, soit directement par un affichage règlementé, soit en permettant à des technologies d'assistance d'intervenir et d'adapter le contenu. Cependant ces normes et recommandations, qu'elles portent sur le contenu ou bien sur les outils de conception et développement, ne font l'objet d'aucune obligation d'application surtout lorsqu'il s'agit d'interfaces conçues par des particuliers. De ce fait, un grand nombre de ressources Web restent encore aujourd'hui inaccessibles.

3.3 Les technologies d'assistance

Au-delà des normalisations, des standards et des recommandations, des outils d'assistance existent et permettent d'aller plus loin. Il existe plusieurs types d'outils et certains d'entre eux sont commerciaux tandis que d'autres sont libres, gratuits et parfois même *open source*. On trouvera par exemple des outils comme les *magnifiers* d'écran qui permettent d'adapter de manière générale l'écran d'un ordinateur à l'aide d'une application locale. D'autres outils plus spécialisés proposent des actions plus spécifiques au Web. Certains systèmes d'exploitation intègrent d'origine des outils d'assistance, simplifiant leur utilisation en évitant à l'utilisateur d'avoir à prendre en charge leur installation et leur mise en route.

3.3.1 Les *magnifiers* d'écran

Les *magnifiers* d'écran aussi appelés agrandisseurs d'écran permettent d'une part comme leur nom l'indique d'utiliser une fonction de zoom (loupe) et d'autre part d'appliquer des filtres de couleurs. Ce type d'outil d'assistance est très connu dans le monde de la déficience visuelle et est largement utilisé dans le monde entier.

L'agrandissement (loupe)

La fonction d'agrandissement est en quelque sorte une loupe électronique. Elle permet d'agrandir une portion de l'écran afin d'augmenter sa visibilité. Cette fonctionnalité se décline en plusieurs modes d'affichage du zoom (cf. Figure 3.1).

Lorsque l'on utilise la fonctionnalité de zoom d'un agrandisseur d'écran, il est parfois possible de choisir entre plusieurs types d'affichage du zoom. Par exemple, ce peut être sous la forme d'une barre qui vient se placer sur le bord de l'écran. Il s'agit d'une zone fixe dans laquelle se trouve un agrandissement de la zone se trouvant autour du pointeur de la souris. Cette zone fixe peut être dimensionnée en fonction des besoins. À la différence de la barre fixe, le mode de grossissement suivant le pointeur est dynamique. Une zone de grossissement suit le pointeur un peu comme si on avait une loupe dans la main et qu'on la faisait circuler au dessus d'un document. La dernière méthode, qui est aussi la plus recherchée par les personnes déficientes visuelles, est le grossissement de type *plein écran*. Cette méthode d'affichage projette l'agrandissement d'une zone de l'écran sur l'intégralité de l'écran. Il suffit ensuite de déplacer cette zone d'agrandissement pour parcourir l'intégralité de l'écran (affichage d'origine). D'une certaine manière, l'écran entier devient la loupe.

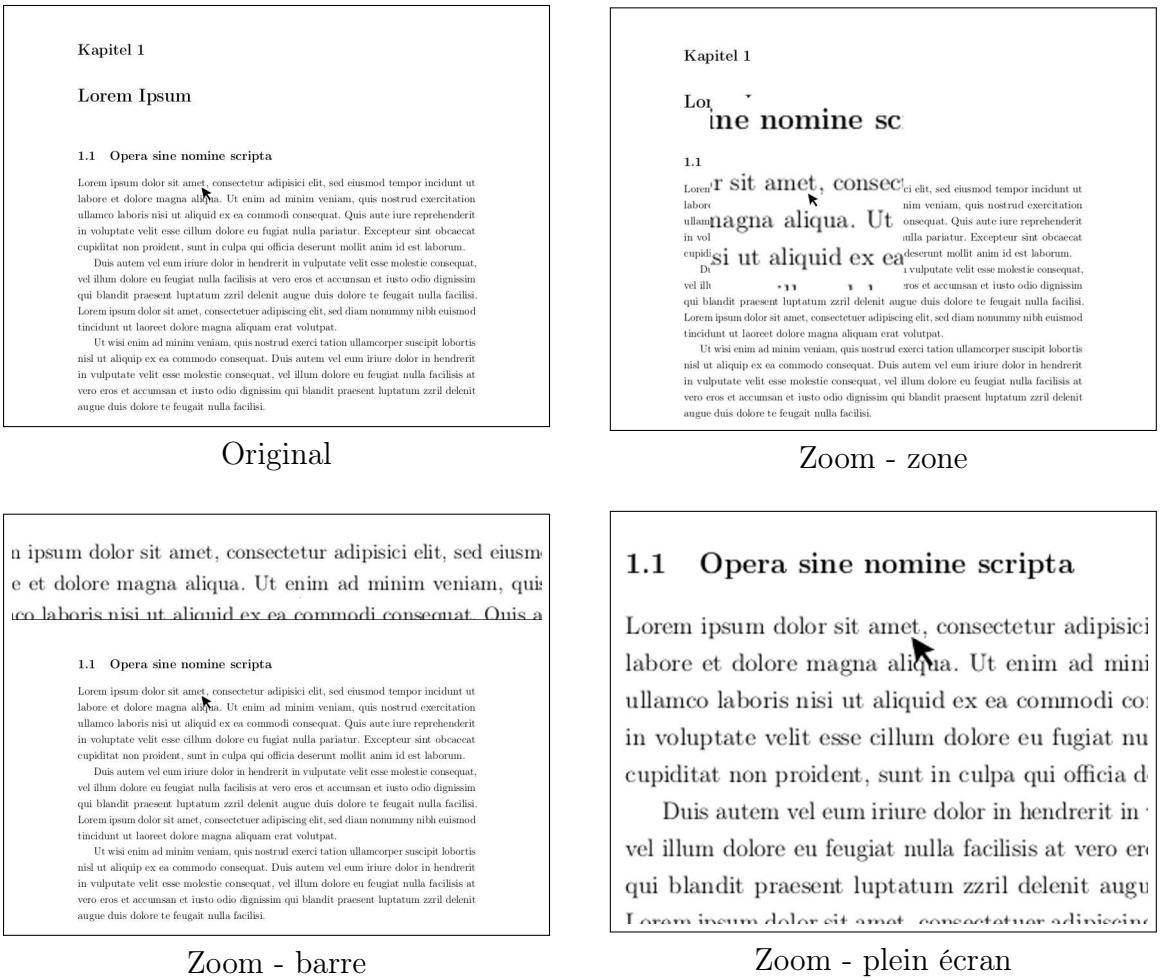


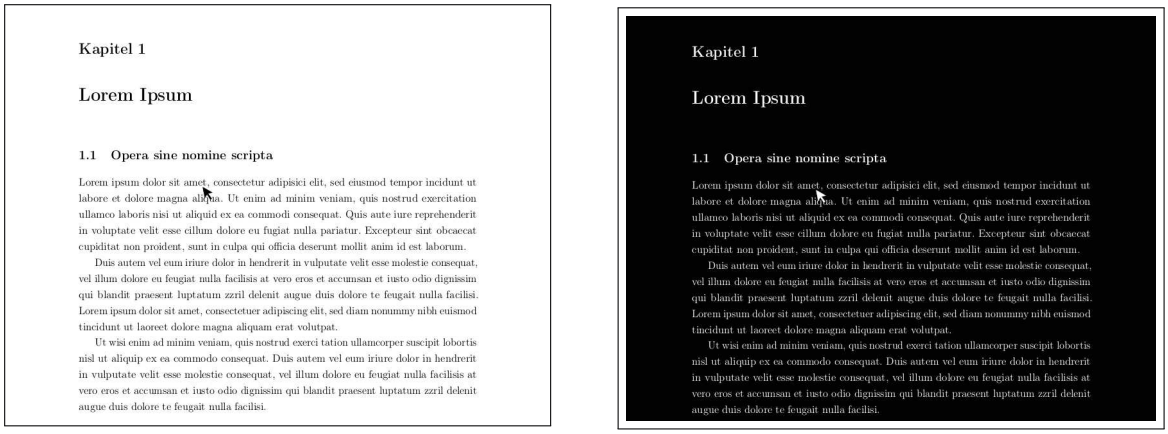
FIGURE 3.1 – Modes de zoom avec les agrandisseur d'écran

Même si l'affichage en mode plein écran est un des plus recherchés, il a l'inconvénient d'empêcher une vue d'ensemble de l'écran d'origine, puisque tout l'écran est utilisé pour afficher l'agrandissement. Certains *magnifiers* d'écran fournissent une fonctionnalité permettant d'enlever complètement le zoom et de le remettre ensuite, le temps d'avoir une vue

d'ensemble. L'affichage du zoom à l'aide d'une barre au bord de l'écran a l'inconvénient de réduire l'espace de travail et de forcer le déplacement des fenêtres. Il arrive régulièrement que des boutons d'actions comme celui de la fermeture des fenêtres (lorsque la barre est en haut) se retrouvent sous la barre et ne soient donc pas utilisables. De plus, comme pour la zone qui suit le pointeur, beaucoup de mouvements sont induits par le mouvement du pointeur tout au long de la lecture. Ces mouvements réguliers peuvent davantage augmenter la fatigue visuelle.

Les filtres de couleurs

Les agrandisseurs d'écran proposent également des filtres de couleurs. On trouvera par exemple des filtres noir et blanc, des filtres niveaux de gris, des filtres niveaux de bleu, rouge, vert, etc. mais le filtre le plus utilisé et le plus répandu est celui d'inversion des couleurs. Les filtres d'inversion de couleurs permettent de remplacer chaque couleur par sa couleur complémentaire (effet négatif). La figure 3.2 montre l'application de ce filtre sur un texte d'exemple affiché à l'écran. L'application de ce filtre sur une page Web est similaire. Pour un exemple sur une page Web réelle, la figure 4.7 donne un exemple de l'application de ce filtre sur une page du SIGACCESS.



Original Couleurs inversées

FIGURE 3.2 – Filtre d'inversion de couleur sur du texte

L'arrière-plan qui est blanc devient noir par l'application de ce filtre. Ceci a l'avantage de réduire l'émission de lumière due à l'arrière-plan clair. Sur le même principe, le texte qui était noir passe ainsi au blanc. Sur l'exemple de la figure 4.7, on peut également voir que toutes les couleurs sont remplacées par leur couleur complémentaire (diamétralement opposée sur le cercle chromatique de la figure 2.10). On constate que sur l'exemple textuel (cf. Figure 3.2) le pointeur de la souris se trouve lui aussi affecté par le filtre d'inversion de couleurs. Ce comportement dépend de l'application (agrandisseur d'écran) utilisée. Selon le fonctionnement de l'agrandisseur d'écran et le niveau auquel il est implanté dans le système d'exploitation, certains éléments comme le pointeur peuvent être ignorés par les filtres.

Le pointeur et le curseur

Le pointeur et le curseur sont deux éléments mobiles qui posent souvent des problèmes de suivi. Certaines de ces applications proposent une solution de compensation par une modification de la taille, de la couleur, de la forme, en ajoutant un cercle autour du pointeur, en faisant apparaître une croix à l'écran représentant la position horizontale et verticale du pointeur et du curseur.

La distinction entre le pointeur et le curseur existe, le pointeur représente la flèche que l'on manipule depuis la souris tandis que le curseur représente la barre clignotante qui indique la position dans un traitement de texte.

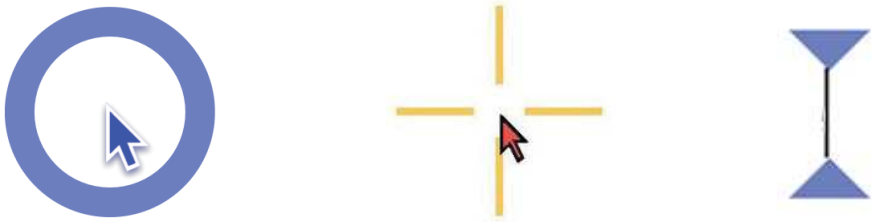


FIGURE 3.3 – Amélioration du pointeur et du curseur

Changement de contexte de lecture

Cette technique est surtout utilisée lors de navigations sur des sites Internet. En effet, selon les pages sur lesquelles l'utilisateur est amené à chercher des informations, il peut avoir des difficultés de lecture dues à un usage important de teintes différentes ou bien à une surcharge d'informations. Le changement de contexte de lecture consiste tout simplement à rediriger le flot de données du site vers une application externe permettant de l'afficher plus confortablement. Cette sélection se réalise avec la souris en traçant un cadre englobant la partie à acquérir. Cette zone du site sera donc extraite et affichée dans une simple fenêtre, sans mise en forme, sans les couleurs de fond ni de texte, c'est simplement un affichage de texte brut sur lequel le magnifier devient très efficace. Ces petits lecteurs externes ne sont pas apparus dès l'origine des agrandisseurs d'écran. D'ailleurs, il ont toute légitimité à exister en dehors et ne sont pas forcément liés à la basse vision.

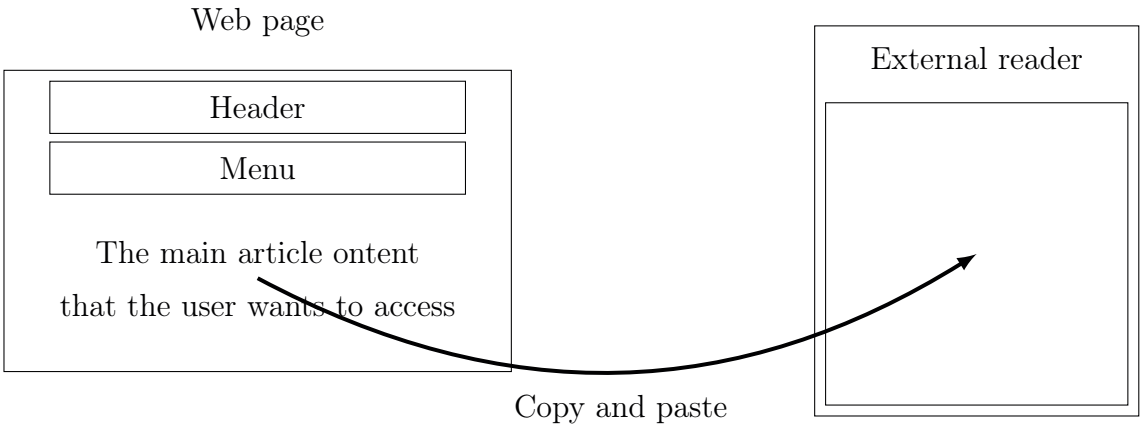


FIGURE 3.4 – Lecture du texte dans un lecteur externe

3.3.2 Intégration dans les systèmes d'exploitation

Les agrandisseurs d'écran ainsi que les outils d'accessibilité en général ont de plus en plus tendance à être intégrés dès l'origine dans les divers systèmes d'exploitation. De cette manière, l'utilisateur n'a pas de manipulation à réaliser afin d'obtenir et d'installer ces outils. Manipulations qui peuvent être évidement compliquées sans ces mêmes outils d'assistance. L'intervention d'une tierce personne peut par conséquent être évitée.

Plates-formes fixes

Les grands groupes comme Microsoft et Apple intègrent entre autres un agrandisseur d'écran. Microsoft a choisi d'intégrer un agrandisseur d'écran dans son système d'exploitation depuis de nombreuses années mais c'est seulement depuis Microsoft Windows 7 que ce dernier intègre la fonctionnalité de zoom plein écran et permet enfin une utilisation assez confortable. Il reste néanmoins très simpliste. Il ne propose qu'un seul mode de traitement des couleurs (négatif) et ne permet pas l'amélioration du pointeur et du curseur. De son côté, Apple fournit un agrandisseur d'écran un peu plus complet avec quelques options supplémentaires. De leur côté, les distributions Linux sont un peu divisées entre elles. Certaines comme Ubuntu intègrent d'origine depuis de nombreuses années un agrandisseur d'écran. D'abord par le biais du compositeur de fenêtres Compiz, qui permettait une très grande flexibilité, puis Ubuntu a développé son propre agrandisseur d'écran compatible avec leur nouvelle interface. Malheureusement ce dernier est encore loin d'être vraiment au point et utilisable pour une utilisation quotidienne. D'autres distributions n'intègrent pas d'agrandisseur d'écran à l'origine mais sont compatibles avec Compiz, et permettent une installation de l'agrandisseur d'écran très simplement. Debian, sur lequel Ubuntu est basé, intégrait par le passé l'agrandisseur d'écran. Debian devrait très certainement encore l'intégrer avec sa prochaine version majeure. Cette fonctionnalité était et devrait être assurée par Compiz.

Plates-formes mobiles

Concernant les terminaux mobile (smartphones et autres), les choses ont beaucoup évolué ces dernières années. Apple fournit un agrandisseur d'écran sur ses terminaux mobiles. Ces derniers sont déjà installés à la sortie d'usine. Google quant à lui, n'offre que depuis peu un agrandisseur d'écran réellement utilisable au quotidien. Le niveau atteint par Google sur son agrandisseur d'écran ne permet plus de dire qu'Apple ou Google est meilleur sur ce point que son concurrent. Microsoft embarque également dès la sortie d'usine un agrandisseur d'écran sur ses terminaux mobiles. Ces derniers bénéficient de Windows 8 qui intègre un agrandisseur d'écran. Un système d'exploitation encore tout jeune, qui est présent depuis quelques années sur les marchés internationaux, est en train de développer ses propres outils d'assistance embarqués. Firefox OS est un système d'exploitation pour terminaux mobile (il tend fortement à s'étendre). Un filtre d'inversion de couleurs est déjà présent sur les versions récentes de l'OS et des fonctionnalités de zoom devraient arriver dans des versions futures.

Les sections suivantes abordent un autre type d'outils et de méthodes d'adaptation. Ces outils sont orientés vers le Web et offrent donc un plus grand potentiel d'amélioration du contenu Web.

3.3.3 Les Outils de gestion des styles

Manipuler les feuilles de style est une manière certainement plus adaptée à l'adaptation des pages Web que l'utilisation d'outils externes aux navigateurs. Ces outils externes ne voyant la page Web que sous la forme d'une image, les éléments de la page sont dans ce cas uniquement des suites de pixels sans aucun sens. Les feuilles de style permettent de définir le style de chaque élément de la page, en particulier suivant son type. Les feuilles de style peuvent être modifiées de différentes manières. La façon dont on va intervenir permettra d'adapter plus ou moins les pages Web.

Option d'accessibilité des navigateurs Web

Une grande partie des navigateurs Web, pour peu qu'ils ne soient pas trop anciens, possède des options d'accessibilité. Parfois celles-ci ne sont accessibles qu'après installation d'une extension sur le navigateur. Dans ces options, il est par exemple possible de manipuler de manière plus ou moins transparente les feuilles de styles des pages Web.

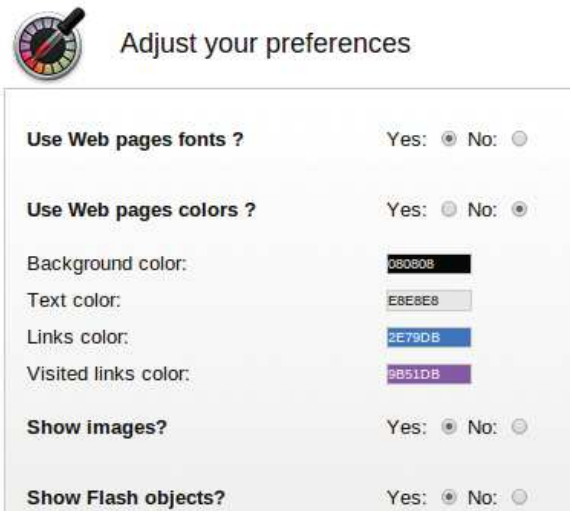


FIGURE 3.5 – Changement de couleurs par la feuille de style (Google Chrome)

Ces fonctionnalités d'accessibilité proposent plusieurs options. La figure 3.5 donne un aperçu des options d'accessibilité disponibles sous Google chrome. Les autres navigateurs Web utilisent des options très similaires.

Suppression de la feuille de style

Il est souvent possible de désactiver complètement la feuille de style afin de ne conserver que le contenu de la page sans sa mise en forme. Dans la figure 3.5 cette option n'est pas visible, cette fonctionnalité doit pouvoir être assurée par une extension du navigateur. Cette fonctionnalité est utilisée par certaines personnes malvoyantes afin de linéariser le contenu. Le fait de rendre le contenu linéaire réduit les difficultés de repérage dans la page. Cela permet également de limiter les textes ou éléments que l'on pourrait oublier car ils sont placés dans une zone mal contrastée ou isolée dans la page. La suppression de la feuille de style a pour effet de ramener tous les éléments de la page sur la gauche de cette dernière et de les placer tous les uns en dessous des autres. Cette approche est un peu « radicale » pour éviter les difficultés dues à la charte graphique choisie par le développeur. Les options d'accessibilité proposent également de ne pas désactiver complètement la feuille de style mais plutôt de la modifier.

Surcharge statique de la feuille de style

Lorsque l'on active les options d'accessibilité pour modifier les feuilles de style, il suffit de renseigner des couleurs pour chaque type d'éléments comme le texte ou les liens afin qu'ils soient modifiés sur toutes les pages. La feuille de style ainsi configurée par l'utilisateur viendra remplacer (surcharger) la feuille de style du site sur lequel il navigue. L'application de cette dernière est donc automatique et directe (aucun calcul n'est nécessaire). Dans certains cas, il est aussi possible de fournir sa propre feuille de style au lieu de configurer les éléments un à un. L'avantage de la surcharge de la feuille de style est qu'il est possible de conserver une partie du contexte initial de la page, notamment au niveau de la disposition des éléments sur la page. Des fonctionnalités autres que les couleurs sont également disponibles et permettent de choisir si on veut afficher ou non les images de la page, si on veut conserver ou non la famille de polices originale. Dans les options d'accessibilité de Google Chrome, il est également possible de désactiver les objets JavaScript. Cette fonctionnalité est souvent disponible dans les navigateurs Web mais pas en tant qu'option d'accessibilité. Dans cet exemple, elle a aussi été ramenée dans ce panneau.

Jusqu'ici la surcharge de la feuille de style se fait de manière statique. C'est-à-dire que les couleurs sont choisies à la configuration (ou une feuille de style est fournie) et lorsqu'une nouvelle page est chargée, la feuille de style est transformée de manière automatique et di-

recte. Des travaux de recherche tentent de rendre ce processus plus dynamique et d'adapter les feuilles de style en prenant en compte un peu plus d'éléments. La section suivante détaille plusieurs travaux de recherches autour de l'adaptation des pages Web, et qui tentent d'apporter des améliorations plus pertinentes que la manipulation des feuilles de style à partir des options du navigateur.

3.4 Travaux de recherche sur l'adaptation de pages Web et les préférences utilisateur

Un certain nombre de travaux de recherche se préoccupent de l'accessibilité au Web. Un travail assez conséquent, présenté dans [Col08], propose à la fois des outils destinés au développeur de pages et des outils destinés aux utilisateurs finaux (internautes). Parmi ces outils, on peut noter un générateur de plan de site, un générateur de braille abrégé, un clavier virtuel adaptatif, ou encore un réorganisateur des blocs d'information d'une page web pour un accès rapide au contenu pertinent. Des algorithmes à fourmis artificielles sont utilisés notamment pour l'optimisation du clavier virtuel et le générateur de plan de site web.

Dans cette section nous allons nous focaliser plus précisément sur différents travaux de recherche autour de la transformation (transcodage) de pages Web. Plusieurs approches sont étudiées et des cibles différentes peuvent être possibles par exemple, les travaux de recherche peuvent être destinés soit à l'utilisateur final soit à un outil, soit au développeur etc. Les sections suivantes organisent ces travaux en fonction d'un certain nombre de critères, selon leur destinataire, leur mode d'implantation, selon que l'approche inclut ou non un démonstrateur, etc.

3.4.1 Architecture de transformation, destinataires et but

Les travaux de recherche menés autour de l'accessibilité des pages Web s'orientent vers des outils de natures différentes [MSPM12]. L'implantation peut être réalisée du côté du client (sur la machine locale). Par exemple, [San09] est un plugin mozilla, [MSPM12] est un script greaseMonkey, ou [IBR10] est un module du navigateur HearSay aural. L'implantation peut également se faire par le biais d'un proxy [MAMS14] ou bien du côté du serveur (hôte) [BKL$^+$06] sous la forme d'une page ou d'un service Web.

Ces outils peuvent être destinés à divers publics ou bien à d'autres outils. Certains outils sont à destination des utilisateurs finaux, par exemple les outils de [San09, MSPM12, MGS02, TCH$^+$02, AMS14]. Certains autres peuvent être destinés à aider le développeur [MGS02] tandis que d'autres encore apportent des améliorations permettant aux outils d'assistance de mieux fonctionner par la suite [LBH08, LHB09, IBR10]. La frontière entre ces outils n'est pas si claire, certains d'entre eux ayant plusieurs destinataires [BL07, MGS02].

La table 3.1 compare les travaux les plus marquants selon ces deux critères.

TABLE 3.1 – Comparaison - Architecture et destinataires

Article	Côté client	Proxy	Côté serveur Page Web ou service	Pour le client	Pour le développeur	Pour un outil d'assistance
[FMPS14a]	✗			✗		
[MSPM12]	✗			✗		
[San09]	✗			✗		
[BL07]	✗	✗	✗	✗	✗	
[TCH+02]	✗			✗		
[BKL+06]		✗	✗	✗		
[MGS02]				✗	✗	
[LBH08]		✗				✗
[LHB09]		✗				✗
[IBR10]	✗					✗
[TBM08]					✗	
[MAMS14]		✗		✗		

3.4.2 Éléments traités, reconnaissance automatique, cible d'application, partie transformée et conservation du contexte originel

La table 3.2 présente une comparaison des travaux existants selon les critères des éléments considérés dans une page.

On peut considérer plusieurs façons d'aborder le problème de l'adaptation de pages Web pour les déficients visuels. Les travaux de recherche s'orientent selon des axes différents et tentent d'adapter des éléments différents dans la page. Certains d'entre eux considèrent uniquement des éléments basiques comme un texte spécifique, des couleurs ou des arrière-plans [San09, TCH+02]. D'autres se focalisent sur des structures logiques de plus haut niveau ou sur le rôle des éléments comme par exemple des listes, des menus, des sections ou des formulaires [LBH08, LHB09, IBR10]. Par exemple des travaux de recherche tentent d'associer les labels aux champs de formulaires auxquels ils correspondent [IBR10]. Cette association est réalisée à l'aide d'une approche par optimisation probabiliste. L'association correcte de ces labels aux champs leur correspondant permet entre autres d'améliorer la lisibilité du document lorsque l'on y accède à l'aide d'un lecteur d'écran.

En parallèle, des approches plus évoluées proposent leur propre méta-modèle ou langage [MGS02, LBH08] dans le but de rendre possible la manipulation d'abstractions de haut niveau. Les approches qui manipulent des éléments de haut niveau nécessitent souvent une phase de reconnaissance de ces éléments dans la page. Cette phase peut être réalisée par reconnaissance de motifs mais elle peut également être réalisée par analyse du code source de la page [MGS02].

Pendant la phase de transcodage de la page, les éléments sur lesquels portent les modifications sont souvent des éléments HTML [LBH08, TCH+02] mais peuvent être aussi des règles CSS [BL07, MSPM12]. Les éléments modifiés peuvent aussi être sous certaines

conditions des scripts (par exemple en JavaScript) [MSPM12, San09].

Dans d'autres travaux de recherche [LHB09], le contenu des pages HTML est enrichi avec des règles de navigation CNR (Content Navigation Rules) en introduisant des attributs ARIA dans les balises des pages. Cet ajout de balises permet une meilleure accessibilité des interfaces Web par les lecteurs d'écran et augmente également la qualité de la navigation au clavier. D'autres travaux se consacrent à l'ajout de texte alternatif sur les images dans la page [BKL+06]. La détermination du texte alternatif est réalisée soit par reconnaissance optique des caractères présents sur l'image, soit en décodant l'URL de cette image.

L'ampleur des modifications dépend de l'approche, certaines considèrent la page Web dans son intégralité et apportent des modifications sur l'ensemble de la page [San09] (approches globales) tandis que d'autres se concentrent sur une partie spécifique (champ d'application des modifications) [FMPS14a, FMPS14b, MPS13] (approches locales).

En ce qui concerne le respect du contexte original de la page (couleur, contenu, disposition, etc.), les travaux de recherche sont assez divisés. Dans certains travaux de recherche [FMPS14a, FMPS14b, MPS13] les auteurs appliquent des modifications qui ne touchent qu'une partie de la page (modifications locales). De ce fait le contexte orignal des parties non modifiées n'est pas altéré. À notre connaissance, les autres approches ne tiennent pas réellement compte de cet aspect et transcodent les pages simplement en fonction des préférences de l'utilisateur.

Table 3.2 – Comparaison - Eléments étudiés et modifiés dans les pages

Article	**Éléments traités**	**Reconnaissance automatique**	**Champ d'application**	**Conservation du contexte original**
[San09]	basiques		global	
[FMPS14a]	basiques		local	
[IBR10]	labels et zones saisies	✗	ajout de labels	
[MGS02]	instances du méta-modèle BML	✗	global	
[BKL+06]	basiques, images		local, ajout de texte alt. images	✗
[BL07]	DOM, CSS		global	
[MSPM12]	DOM, CSS, Scripts		global	
[LBH08]	annot. sémantique CSS		global	sémantique des éléments
[LHB09]	annot. sémantique CSS		global	sémantique des éléments
[TCH+02]	basique, tables		global/local	couleurs derrière les images transparentes
[TBM08]	couleurs	palette de couleurs	global	✗
[MAMS14]	couleurs	✗	global	✗

Concernant cet aspect de conservation du contexte original de la page, notre approche se base sur les préférences de l'utilisateur mais aussi sur les préférences du designer (contexte original de la page). Ceci amène potentiellement des conflits supplémentaires entre les préférences, même si des conflits pouvaient déjà être présents entre les seules préférences de l'utilisateur. À la différence des autres travaux de recherche, les utilisateurs peuvent donner des préférences à la fois sur des éléments de bas niveau, mais aussi sur des éléments de plus haut niveau d'abstraction. Ceci est réalisé par la définition de variables décrivant des paires composées d'un élément (à terme d'une structure) de la page et d'une de ses propriétés potentiellement complexe (taille, couleur, etc.). Les préférences peuvent être n'importe quelle fonction évaluable exprimée sur ces variables et leur domaine de définition.

3.4.3 Profil utilisateur : langage de définition, acquisition, déficience spécifique visée et définition a priori

TABLE 3.3 – Comparaison - Profil de l'utilisateur

Article	Langage de définition	Acquisition	Déficience spécifique	Définition à priori
[San09]	type de défaut de vision (color blindness ou hyposight)	interface de déclaration du type de défaut	color blindness ou hyposight	✗
[FMPS14a]	affectation de valeurs à des éléments de bas niveau	Q-learning pour acq. initiale puis sur actions utilisateurs		
[IBR10]				
[MGS02]				
[BKL+06]			utilisateur de lecteur d'écran	
[BL07]	script			✗
[MSPM12]	IMS ACCLIP	interface de saisie		
[LBH08]			utilisateur de lecteur d'écran	
[LHB09]			utilisateur de lecteur d'écran	
[TCH+02]	affectation de valeurs à des éléments de bas niveau	interface de saisie		
[TBM08]	type de déficience	interface dépendant de la déficience visuelle	principalement daltonisme	✗
[MAMS14]		indication de la déficience	dyschromatopsie	✗

Le profil de l'utilisateur peut se présenter de différentes manières et être plus ou moins présent et pris en compte dans les travaux de recherche. On peut par exemple le trouver sous la forme d'un ensemble de valeurs attendues sur des caractéristiques spécifiques comme par exemple la taille de police de caractères ou bien la couleur du texte [TCH+02, San09, MPS13]. Le profil de l'utilisateur peut soit être défini à l'avance pour des catégories d'utilisateurs connues à l'avance comme par exemple pour le daltonisme ou d'autre défauts dans la perception des couleurs [San09]. Soit ce profil peut être acquis. C'est-à-dire que l'utilisateur saisit ses préférences par le biais d'une interface permettant de choisir des valeurs pour certaines caractéristiques spécifiques [TCH+02, MSPM12]. Dans une autre approche, des tests visuels sont présentés à l'utilisateur et le système en déduit ses préférences par la suite [FS09]. Dans une approche plus avancée, le profil utilisateur est appris à partir des actions de l'utilisateur et d'un algorithme d'apprentissage (Q-learning) [FMPS14a, FMPS14b, MPS13]. La table 3.3 synthétise les approches existantes sur ces critères.

Dans nos travaux de recherche nous ne traitons pas du problème de l'acquisition du profil utilisateur. Cette partie est néanmoins abordée et après analyse du type de besoins réels de l'utilisateur, nous proposons des pistes pour des travaux futurs sur l'acquisition de profil.

3.4.4 Méthode d'adaptation

En fonction de la forme que prend le profil de l'utilisateur, l'étape d'adaptation peut se présenter sous différentes formes. Dans certains cas, l'application d'un profil utilisateur prédéfini peut se faire de manière directe sur la page [San09], ou bien c'est le profil de l'utilisateur appris qui est appliqué directement sur la page [MPS13]. Dans le projet *Cloud4all*, les participants travaillent sur une infrastructure publique inclusive globale (GPII *Global Public Inclusive Infrastructure* qui permettra à un profil utilisateur d'être facilement implémenté pour personnaliser n'importe quel périphérique et contenu.

À la différence de ces travaux de recherche, dans notre cas, le profil de l'utilisateur peut contenir des préférences conflictuelles. Un compromis doit par conséquent être trouvé entre les différentes préférences. Ceci est réalisé en utilisant un algorithme méta-heuristique permettant d'obtenir une bonne approximation. Une approche de ce type est également proposée dans [MAMS14], qui re-colore les textes avec une approche méta-heuristique.

3.4.5 Évaluation, métriques et utilisateurs

La table 3.4 rend compte des travaux sur le volet de l'évaluation, de l'utilisation de métriques dans l'approche et de la présence d'utilisateurs lors de l'évaluation.

Concernant les évaluations, certaines approches proposent également des métriques permettant de juger les résultats obtenus [San09]. Certaines métriques peuvent aussi être définies afin de guider le processus [IBR10]. Certaines approches intègrent dans leurs évaluations des utilisateurs [IBR10], la définition de métriques dans ce cas est donc importante.

Dans notre approche, des fonctions objectif sont utilisées par les algorithmes génétiques

afin de conditionner l'évolution. Ces fonctions objectif peuvent être considérées comme des métriques guidant le processus et leur taux de satisfaction sert à évaluer le résultat. D'autres métriques ont également été utilisées afin de réaliser des expérimentations sur des pages réelles. Ces métriques permettent de comparer les résultats obtenus par plusieurs algorithmes méta-heuristiques. La définition de métriques permettant à des utilisateurs d'évaluer notre approche fait partie des travaux à venir.

TABLE 3.4 – Comparaison - Évaluations

Article	**Évaluations**	**Métriques**	**Utilisateurs**
[San09]		lisibilité, contraste	
[FMPS14a]	✗		✗
[IBR10]	✗	guidage de l'optimisation	✗
[MGS02]		nombre d'images, redondance d'image, nombre d'images avec texte alternatif	
[BKL+06]	✗	mesure de la pertinence des textes alternatifs	✗
[BL07]			
[MSPM12]	✗		✗
[LBH08]	✗		✗
[LHB09]			
[TCH+02]			
[TBM08]	✗	couleurs (distance, contraste)	
[MAMS14]	✗	couleurs	

3.4.6 Démonstrateur et corpus de données fourni

Plusieurs auteurs mettent en place des démonstrateurs ou prototypes. Ces derniers permettent entre autres de valider plus facilement une approche. Certains mettent à disposition ces démonstrateurs et peuvent donc être testés. Ils peuvent se présenter sous différentes formes comme des plugins pour un navigateur [TCH+02, San09]. Pour ce qui est des codes source, il est bien plus difficile de les trouver. Ces derniers ne sont pas forcément mentionnés même s'ils sont mis à disposition.

De notre côté, nous décrivons dans ce document les parties clés de notre démonstrateur, de plus les données sources utilisées afin de calculer l'adaptation des pages sont également fournies en annexe.

3.5 Conclusion

Nous avons parcouru différents travaux de recherche autour de l'adaptation de pages Web. Ces travaux ne ciblent pas exclusivement la basse vision, certains d'entre eux sont plutôt orientés vers la non-voyance. Cependant la façon dont le profil utilisateur est défini et les méthodes de transformation utilisées peuvent être appliquées à d'autres domaines.

Les articles traitent de diverses manières de créer ou d'obtenir un profil utilisateur et des types de préférences qu'ils prennent en compte. Les préférences manipulées se limitent

à des éléments basiques de la page et prennent souvent la forme d'une association de valeurs pour un ensemble de caractéristiques définies. De ce fait, la transformation est très simple puisqu'il n'y a pas de conflit dans les valeurs données. Sur ce point, notre approche tente d'aller plus loin en prenant en compte des préférences plus complexes et sur des éléments de plus haut niveau. Ces dernières peuvent amener à des dépendances (relations) entre les éléments de la page (par exemple un contraste entre deux éléments). Des dépendances et conflits pouvant intervenir, la méthode d'adaptation doit pouvoir les prendre en compte. À la différence des travaux de recherche existants, nous utilisons une méthode de résolution à base de méta-heuristiques permettant de trouver un bon compromis entre les préférences lorsqu'elles sont conflictuelles. De plus nous intégrons également un aspect de conservation du contexte original de la page que ce soit sur les parties transformées ou non.

Diverses métriques ont été mises en place, d'une part pour conditionner l'évolution de notre algorithme de résolution (fonctions objectif), d'autre part pour mener des expérimentations et comparer différents algorithmes. L'utilisateur final sera associé très prochainement à nos travaux de recherche de manière à valider et adapter le processus de transformation au fil de l'avancement du projet.

Le chapitre suivant détaille plus précisément plusieurs difficultés qui peuvent être rencontrées par les utilisateurs ayant une basse vision et montre comment les technologies d'assistance existantes tentent de répondre aux besoins et quelles sont leurs limites.

Chapitre 4

Problématiques de visualisation des interfaces Web

Préambule

Alors que l'accessibilité des applications locales est relativement bien traitée, il en est tout autrement concernant les pages Web. Ce chapitre met en avant les diverses raisons qui font que les pages Web amènent bien plus facilement à des problèmes d'accessibilité que les applications locales. Le problème réside principalement dans la liberté offerte aux concepteurs et développeurs de pages Web, mais pas uniquement. Divers exemples permettent de mettre en avant des problèmes concrets rencontrés par des personnes ayant une basse vision mais aussi des problèmes qui peuvent être rencontrés par n'importe qui dans un contexte particulier. Ce chapitre introduit aussi brièvement des solutions pour compenser les difficultés rencontrées. Ces solutions seront détaillées plus précisément dans les chapitres suivants. Ce chapitre est construit sur la base d'entretiens informels menés auprès de personnes malvoyantes (avec diverses pathologies) au sein d'une association.

Sommaire

4.1 Problèmes sur la page d'origine **65**
4.1.1 Éblouissements (brillance importante) 65
4.1.2 Contraste faible 66
4.1.3 Texte au format image 67
4.1.4 Images d'arrière-plan 68
4.1.5 Ordre du contenu 69
4.1.6 Le daltonisme 69
4.2 Utilisation des technologies d'assistance **70**
4.2.1 Éblouissements (brillance importante) 71
4.2.2 Contraste faible 72
4.2.3 Images d'arrière-plan 72
4.2.4 Ordre du contenu 74
4.2.5 Le daltonisme 74
4.3 Conclusion **74**

Les divers outils d'assistance, spécialisés ou non pour le monde du Web, étudiés dans le chapitre 3 « État de l'art », apportent des améliorations notables. Cependant, ils ne sont pas forcément utiles à toutes les personnes déficientes visuelles. En effet, du fait de leur conception, il ne sont pas toujours en mesure de traiter tous les cas de manière cohérente et pertinente. Les sections suivantes mettent en évidence certains des problèmes fréquemment rencontrés par des personnes déficientes visuelles. Le choix des problèmes montrés dans ce chapitre a été fait selon des retours de personnes en situation de basse vision.

4.1 Problèmes sur la page d'origine

Comme on l'a déjà vu dans le chapitre 3, peu de pages Web respectent complètement les standards du W3C. Toutefois même si les pages sont totalement conformes aux standards, celles-ci ne peuvent être correctement accessibles pour toute personne. Des difficultés de natures variées et inhérentes aux pathologies peuvent survenir et ce, indépendamment du respect ou non des standards et recommandations d'accessibilité.

4.1.1 Éblouissements (brillance importante)

Les personnes atteintes de photophobie sont très sensibles aux sources de lumière importante. Si l'image projetée est claire, un écran d'ordinateur peut générer assez de lumière pour éblouir complètement une personne. Un grand nombre de pages Web sont sur fond blanc. Cette tendance dépend du thème de site (e-commerce, voyage, loisir, etc.) sur lequel on navigue. Mais le fond blanc (ou très clair) fait partie des plus répandus.

Une page Web peut tout à fait respecter les standards tels que ceux du W3C ou du RGAA et avoir un arrière-plan blanc sur toute la page. Malheureusement ceci ne convient pas du tout aux personnes qui sont très sensibles à la lumière et donc facilement éblouies.

Le texte devient difficilement voire complètement illisible. Cette gêne n'apparaît pas uniquement lorsque tout le fond de la page est blanc. il arrive également que des difficultés de lecture surviennent même s'il ne s'agit que d'un petit bloc de texte sur un fond blanc. Le lecteur peut également être gêné si un élément clair se trouve à proximité d'un texte sans pour autant que ce soit l'arrière-plan de ce dernier. Nous avons pris pour exemple le blanc comme source de problème parce qu'il s'agit de la couleur la plus lumineuse, mais le problème peut arriver avec toutes les couleurs lumineuses. La figure ci-contre (Figure 4.1) donne un aperçu de ce qu'une gêne due à un arrière-plan lumineux peut donner. Il ne s'agit que d'une représentation donnée à titre d'exemple. Lorsqu'une personne est gênée par un arrière-plan trop lumineux, le texte a tendance à paraître plus fin. La couleur de l'arrière-plan déborde sur la lettre ce qui donne des contours flous. Si l'éblouissement devient trop important, les lettres peuvent entièrement disparaître. L'impact peut alors aller d'une simple réduction légère de la vitesse de lecture à une impossibilité totale de lecture.

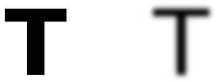


FIGURE 4.1 – Effet d'éblouissement par l'arrière-plan (lettre gauche : pas d'éblouissement, lettre droite : éblouissement)

4.1.2 Contraste faible

Le contraste est ce qui permet de distinguer deux éléments entre eux. Plus le contraste est élevé et plus la discrimination sera facile. Sur une page Web, il est important d'avoir un bon contraste entre le texte et l'arrière-plan afin d'avoir un confort de lecture optimal. Des pathologies visuelles affectent la perception des couleurs et par la même occasion la perception des contrastes. Qu'il s'agissent des contrastes de luminance ou de couleurs (cf. Section 2.4.2), des contrastes trop bas peuvent empêcher la lecture.



FIGURE 4.2 – Exemples de contrastes faibles

Certains concepteurs ou développeurs de site Web utilisent des ensembles de couleurs qui offrent des contrastes entre le texte et l'arrière-plan trop faibles. Des standards permettent de normaliser le taux de contraste de brillance minimal nécessaire entre le texte et son arrière-plan immédiat, mais cela n'est bien sûr pas suffisant pour tout le monde. Les personnes qui sont atteintes de pathologies à un stade avancé sont souvent confrontées à ce problème. Il n'est pas si rare de trouver des textes verts sur fond gris moyen, ou même jaunes sur fond blanc. il est parfois même difficile pour des personnes valides de les lire selon le support de visualisation utilisé. La figure 4.2 donne quelques exemples de contrastes faibles entre le texte et l'arrière-plan immédiat. Le contraste de luminance (ou brillance) est le type de contraste le plus utilisé. Le W3C donne une méthode de calcul et l'utilise pour définir ses standards. Quelques exemples de cas rencontrés lors de navigations diverses sur le Web sont présentés en figures 4.3 et 4.4. On peut y trouver des taux de contraste relativement faibles. Sur certains exemples, le texte peut même devenir complètement illisible pour un grand nombre de personnes selon le support sur lequel ils sont visualisés. Par exemple, avec un vidéo-projecteur qui a bien moins de contraste qu'un moniteur à dalle IPS, la perception risque bien d'être difficile pour beaucoup de personnes.

Les figures 4.3 et 4.4 présentent des exemples concrets avec les sites Web de l'école nationale d'architecture de Montpellier et de l'école nationale d'architecture de Paris. Sur le premier exemple (École nationale d'architecture de Montpellier [1]), on constate dans les menus une forte utilisation des couleurs. Sur l'exemple, l'onglet ouvert donne le sous-menu sur un arrière-plan d'un vert lumineux. Le texte blanc se trouvant dessus est par conséquent peu contrasté avec l'arrière-plan. Les autres menus sont dans le même cas (mis à part le premier) avec des arrière-plans bleu clair, lavande, ou rouge.

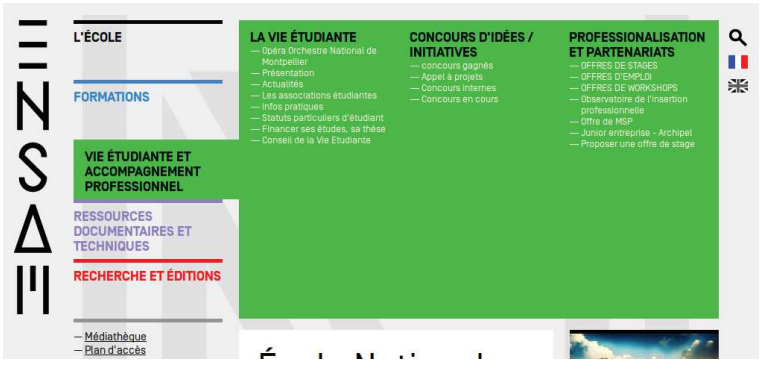


FIGURE 4.3 – École nationale de Montpellier (Menu)

1. http ://www.montpellier.archi.fr/, (20 août 2015)

Pour ce qui est du site Web de l'école nationale d'architecture de Paris [2], le menu de gauche peut facilement poser des problèmes de lisibilité. En effet, le texte se trouve en blanc sur un fond gris clair. Le contraste est par conséquent relativement mauvais. Ce genre de contraste entre le texte et l'arrière-plan peut poser des problèmes de lecture à beaucoup de personnes sans qu'elles aient nécessairement une déficience visuelle avérée. En fonction du support qui est utilisé pour consulter cette page la distinction du texte par rapport à l'arrière-plan peut être très difficile. De plus, si le contexte extérieur n'est pas favorable, par exemple avec un éclairage important, la lecture sera d'autant plus difficile. Dans ce genre de cas, il peut ne plus s'agir de lecture mais d'interprétation à partir des quelques informations que l'on perçoit.



FIGURE 4.4 – École nationale de Paris (accueil)

4.1.3 Texte au format image

Certains sites Web utilisent des images afin de véhiculer des informations textuelles. C'est souvent le cas des sites Web développés avec la technologie Flash ou des technologies similaires, mais on trouve également des sites en HTML qui utilisent ce procédé. Bien que les textes affichés par le biais d'images soit strictement déconseillés par les standards WCAG 2.0, il restent néanmoins toujours une réalité.

Divers problèmes peuvent survenir, notamment lors de la consultation de ces textes par des personnes malvoyantes. Ces textes, affichés par le biais d'images, ne peuvent pas être sélectionnés ni copiés. Certains utilisateurs malvoyants utilisent un outil externe dans lequel ils placent le texte à lire afin de lui enlever sa mise en forme et d'accroître le confort de lecture. Cependant, avec les images, il n'est pas possible de copier le contenu textuel et donc de l'insérer dans ce type d'outils. D'une certaine manière, ces textes ne peuvent être lus que visuellement puisqu'en soi ce n'est pas du texte mais une image. Un lecteur d'écran ne sera donc pas en mesure d'y accéder non plus.

À côté de cela, l'agrandissement de ce texte peut lui aussi poser problème. Si le niveau de grossissement reste faible alors cela n'est pas vraiment problématique mais dès lors que l'on atteint des seuils d'agrandissement un peu plus importants alors la netteté du texte diminue rapidement. Le fait d'agrandir une image quelle qu'elle soit diminue sa qualité (un phénomène dit *de pixélisation* intervient). Lorsqu'il s'agit d'agrandir une image pour lire du texte alors ce dernier est dégradé ce qui réduit la qualité de lecture.

2. http ://www.paris-malaquais.archi.fr/news-1.html, (20 août 2015)

4.1.4 Images d'arrière-plan



FIGURE 4.5 – Contraste non uniforme (texte sur image)

Outre l'information que peuvent véhiculer les images placées en tant qu'arrière-plan, bien que strictement déconseillées par le WCAG 2.0, ces images peuvent sensiblement réduire le confort de lecture si elles sont insérées au-dessous de contenus textuels. En effet, le texte peut ainsi devenir illisible à certains endroits selon la teinte et l'intensité lumineuse de la zone de l'image sur laquelle il se trouve. La couleur de l'image à l'endroit où se trouve le texte peut par exemple donner lieu à un contraste faible ou à des éblouissements. Ce procédé est souvent utilisé pour la création du bandeau d'entête des pages Web ou bien pour la création de boutons. L'exemple de la figure 4.5 donne un aperçu des problèmes pouvant être rencontrés lorsque l'on affiche du texte au-dessus d'une image. Le site Web eco-écolo.com [3] utilise ce principe de superposition de texte et d'image pour la réalisation de son bandeau d'entête. On y trouve le titre général (du site Web) ainsi qu'une image d'arrière-plan qui est insérée au-dessous de ce titre.

Sur cette page, le contraste de brillance entre certaines parties du texte (titre général) et l'image d'arrière-plan est très faible. Par exemple, le début du mot « Bio » et la fin de « Médecine » se trouvent au-dessus de dessins dont la teinte est proche de celle du texte, ce qui réduit grandement leur lecture. Les lettres se fondent dans l'image. Cet exemple montre le type de constructions (texte sur image) pour lesquelles des problèmes de contraste peuvent arriver. Certains sites Web utilisent des jeux de couleurs pour les textes et des images qui amènent à des contrastes extrêmement réduits.

De plus, bien souvent la méthode utilisée pour le développement de la page et permettant de positionner les images ainsi que les textes ne permet pas de prévoir la superposition. C'est-à-dire que le texte n'est pas positionné de manière fixe (absolue) par rapport à l'image. Autrement dit, le niveau de zoom du navigateur, la famille de police de caractères utilisée ou la résolution de l'écran, sont autant de paramètres qui jouent sur le placement du texte au dessus de l'image. Il n'est pas possible de compter sur le fait que l'image contient un grande portion avec laquelle le texte est bien contrasté puisque l'on ne sait pas sur quelle portion de l'image il va finalement se positionner.

3. http ://eco-ecolo.com/environnement/economiser-electricite-comment-calculer-la-puissance-declairage-de-sa-piece//, (28 septembre 2015)

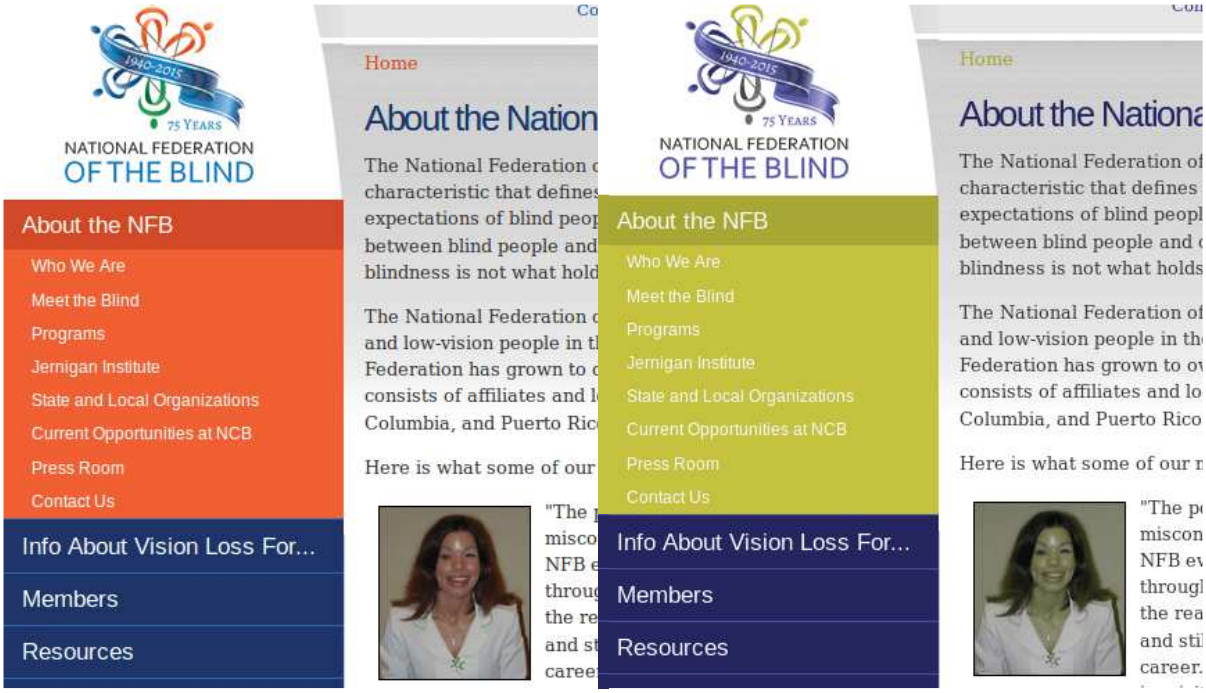
4.1.5 Ordre du contenu

Nous abordons ici un point un peu plus complexe qui dépend fortement de la subjectivité de chacun et de la culture ou de l'expérience de la personne qui accède à la page Web. La réduction de champ visuel provoque l'augmentation du balayage visuel. Un champ visuel étroit réduit la quantité d'informations visibles en même temps. Pour accéder à l'intégralité de la page, il est donc nécessaire de la balayer à la fois verticalement et horizontalement. Plus le champ visuel est réduit et plus le balayage sera important. À cela viennent s'ajouter l'expérience et les habitudes de chacun ainsi que des critères sociaux-culturels. Des études ont été menées dans le but de comprendre comment on découvre une page Web inconnue [BC00]. Une caméra filme le mouvement des yeux et permet de déduire les sections de pages regardées et ce, dans quel ordre et pendant combien de temps. La méthode a l'avantage de pouvoir étudier une personne dans son propre contexte sans réellement avoir d'intrusions. Les résultats de ces travaux montrent largement que la découverte d'une nouvelle page est conditionnée selon un certain nombre de critères notamment culturels. Des divers résultats obtenus, il ressort que la représentation mentale d'une page Web sera d'autant plus facile à construire que les divers éléments se trouvent là où on a l'habitude de les trouver. Par exemple, une page simple avec un menu à gauche et un contenu central sera certainement plus facile à comprendre et mémoriser qu'une page où le menu se trouverait en dessous du contenu principal. Cependant ceci est vrai pour certaines personnes, d'autres personnes ayant une culture différente pourraient être habituées à avoir le menu à droite. Cette habitude peut être liée au sens de lecture de la personne. Les pages Web ayant donc des structurations complexes de l'information peuvent par conséquent être un frein important à l'accès à l'information contenue.

En plus des besoins de contrastes ou de taille de police de caractères spécifique, certains utilisateurs émettent aussi le besoin d'avoir une structuration de l'information différente afin de leur faciliter la lecture. Dans l'étude informelle menée (cf. chapitre 5), une des personnes interrogées a émis le besoin d'une certaine linéarisation de l'information. Ce besoin peut avoir des origines diverses et on ne peut pas l'interpréter de manière objective. Cependant, l'expérience nous a montré que les personnes qui utilisent régulièrement un lecteur d'écran en parallèle ont un certain besoin de linéarisation du contenu. En effet, les lecteurs d'écran permettent d'accéder au contenu d'une page mais l'accès se fait en quelque sorte de manière complètement linéaire, il n'y a plus de notion de positionnement géographique. Seule la hiérarchie des titres et quelques éléments comme les tableaux permettent de se repérer dans la page.

4.1.6 Le daltonisme

Le daltonisme est une pathologie relativement connue du grand public mais dont l'impact sur l'accès à l'information l'est beaucoup moins. Le défaut de perception de certaines couleurs peut entraîner des difficultés de lecture en rejoignant dans certains cas les problèmes rencontrés lorsqu'un contraste entre le texte et l'arrière-plan est trop faible. Même si à la base le contraste entre l'arrière-plan et le texte est assez important, la déformation dans la perception d'une des deux couleurs (arrière-plan ou texte) peut le réduire fortement.



Original Protanopia

FIGURE 4.6 – Effet de la protanopie (type de daltonisme)

Le site Web de la National Federation of the Blind [4], est un exemple de site Web où une personne daltonienne peut être gênée. La figure 4.6 montre la page originale de la NFB (à gauche) et la page telle qu'elle sera perçue par une personne atteinte de protanopie [5] (une des trois grandes formes de daltonisme). On peut voir qu'un certain nombre de couleurs sont touchées et altérées. On constate notamment que la couleur de fond du menu de gauche (partie haute) s'est éclaircie. Le contraste entre le texte du menu et son fond est réduit par rapport à ce qu'il est à l'origine. D'autres formes de daltonisme existent et peuvent poser d'autres problèmes. De manière générale, plus l'information sera transmise par le biais de couleurs et plus les difficultés seront importantes. Dans le cas de schémas ou cartes géographiques, l'utilisation de couleurs comme seul moyen de véhiculer l'information peut donc être problématique.

4.2 Utilisation des technologies d'assistance

Les technologies d'assistance, sous la forme par exemple de logiciels ou d'applications, permettent de réduire certaines difficultés et parfois même de les supprimer complètement. Leurs diverses options permettent à un utilisateur de configurer les transformations afin d'adapter le plus possible le rendu à ses besoins (cf. Chapitre 3). Cependant les transformations apportées par ces outils d'assistance ne sont pas toujours adaptées aux besoins réels de l'utilisateur et ne peuvent résoudre certains cas complexes. Les sous-sections suivantes

4. http ://www.nfb.org, (26 Août 2015)
5. Transformation réalisée avec l'outil ColorTest.it

reprennent les divers problèmes abordés dans la section précédente. Pour chacun de ces problèmes, nous allons montrer des exemples de transformations qui peuvent être effectués par le biais des technologies d'assistance. Nous mettrons ainsi en avant les faiblesses et limites éventuelles de ces outils.

4.2.1 Éblouissements (brillance importante)

Lorsqu'un arrière-plan sur la page Web est trop lumineux, il peut entraîner des éblouissements pouvant gêner ou même empêcher la lecture. Les *magnifiers* d'écran offrent la possibilité d'appliquer des filtres de couleurs afin de palier notamment cette difficulté. Les filtres d'inversion de brillances ou d'inversion de couleurs permettent de faire passer les arrière-plans lumineux vers leur « couleur opposée » pour qu'il deviennent sombres. À l'inverse, les couleurs sombres deviennent alors lumineuses. Sur le principe, cela fonctionne très bien. L'inconvénient principal réside dans la manière d'appliquer les filtres de couleurs. La plupart des *magnifiers* d'écran sont des applications locales qui interagissent avec des couches basses du système d'exploitation. Là où se produisent ces transformations, il n'y a que des images (carte graphique). La zone affectée par les filtres de couleur peut varier d'un outil à l'autre mais dans le meilleur des cas le filtre s'appliquera à une fenêtre entière. Autrement dit la page Web est affectée dans son intégralité. C'est justement l'incapacité d'appliquer un filtre de couleur sur une portion spécifique de la page Web qui est la source de nombreux problèmes.



FIGURE 4.7 – Brillances opposées et filtre d'inversion de couleurs

Dans certains cas, la page Web consultée contient plusieurs parties distinctes qui se trouvent sur des brillances d'arrière-plan diverses. Par exemple une page peut contenir un menu sur fond bleu et un contenu principal sur fond blanc (cf. Figure 4.7). Le texte du menu principal se trouve donc sur un fond sombre propice à une lecture assez confortable, mais ce n'est pas le cas du contenu principal. L'utilisation du filtre d'inversion de couleurs fait passer le fond lumineux du contenu principal vers un fond sombre, ce qui améliore grandement le confort de lecture en réduisant la source principale de lumière. Cependant, le menu qui était au préalable sur un fond sombre se retrouve maintenant sur un fond lumineux.

Il n'est pas rare de se trouver confronté à ce genre de configuration sur les pages Web. D'autant que de nos jours les pages Web de ce type se multiplient très rapidement sur la toile.

4.2.2 Contraste faible

Le problème de contraste est un problème récurrent pour les personnes ayant une basse vision. Les *magnifiers* d'écran permettent d'appliquer des filtres de couleurs pour tenter d'améliorer ce dernier. Il existe plusieurs filtres différents. La figure ci-contre (Figure 4.8) montre un exemple de l'application de divers filtres de couleurs sur une image qui contient un texte coloré sur un arrière-plan coloré. Le contraste initial de (A) est volontairement faible comme on le rencontre régulièrement sur des pages Web. L'inversion de couleurs est certainement le filtre le plus utilisé, il s'agit d'inverser l'ensemble des couleurs de l'image (effet négatif). Le cas B montre l'application de ce filtre sur l'image A. Le texte et l'arrière-plan se retrouvent bien affectés de leurs couleurs opposées (complémentaires). Cependant le contraste entre ce texte et son arrière-plan reste similaire. Cette configuration de couleurs entre les deux éléments n'est pas vraiment améliorée par l'application des filtres de couleurs les plus répandus et même dans certains cas le filtre peut dégrader l'image. Le cas C montre une transformation en niveaux de gris. On observe que le contraste n'est pas non plus amélioré. Le cas D montre l'application d'un filtre de conversion en noir et blanc. Lorsque l'on utilise ce filtre, il y a un paramètre important qu'il faut prendre en compte, c'est le seuil à partir duquel on détermine si la couleur est transformée en noir ou en blanc. Lorsque le texte et l'arrière-plan sont très proches, il se peut que la conversion en noir et blanc fasse passer le texte et le fond du même côté (tous les deux noirs ou tous les deux blancs. Dans le cas D, les deux sont blancs. Par expérience, lorsque l'on utilise ce filtre, ce fameux seuil est loin d'être facilement manipulable et utilisable par l'utilisateur. Il est souvent caché dans des configurations avancées et de plus il est choisi pour toutes les applications futures et n'est pas dynamique.

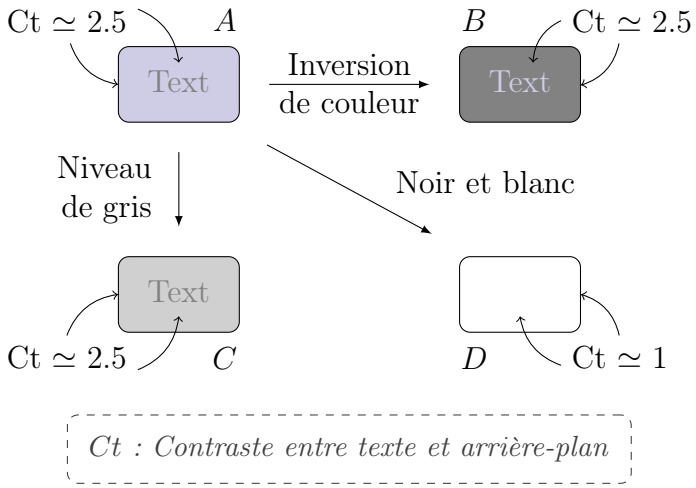


FIGURE 4.8 – Application de filtres

4.2.3 Images d'arrière-plan

Les images d'arrière-plan sont une source récurrente de problèmes car elles peuvent à la fois véhiculer de l'information ou réduire l'accès à l'information qui pourrait se trouver au-dessus. Les filtres de couleurs n'ont que peu d'impact sur ce type de problème. Les navigateurs Web (pour peu qu'il ne soient pas trop anciens) proposent quant à eux des options d'accessibilité permettant notamment de désactiver l'affichage des images insérées en tant qu'arrière-plan. Cette action peut ne pas être sans conséquences sur la transmission de l'information.

Le site Web de la BBC présente un bulletin météorologique sur une de ses pages [6]. Une carte sur laquelle sont situées les différentes villes est affichée afin de sélectionner la ville pour laquelle on veut obtenir le bulletin. Cette carte est construite sous la forme d'un ensemble de liens positionnés au-dessus d'une image d'arrière-plan qui représente la carte en question. Le problème de la désactivation des images d'arrière-plan par le biais des options d'accessibilité du navigateur est que le positionnement des liens reste non affecté. De ce fait cette organisation des liens sur la page ne fait aucun sens si on n'a pas vu la page telle qu'elle est au départ. Dans ce cas de figure, il serait certainement bien plus intéressant pour l'internaute d'avoir une liste ordonnée des villes les unes au-dessus des autres.



FIGURE 4.9 – Perte d'informations par la désactivation des images d'arrière-plan

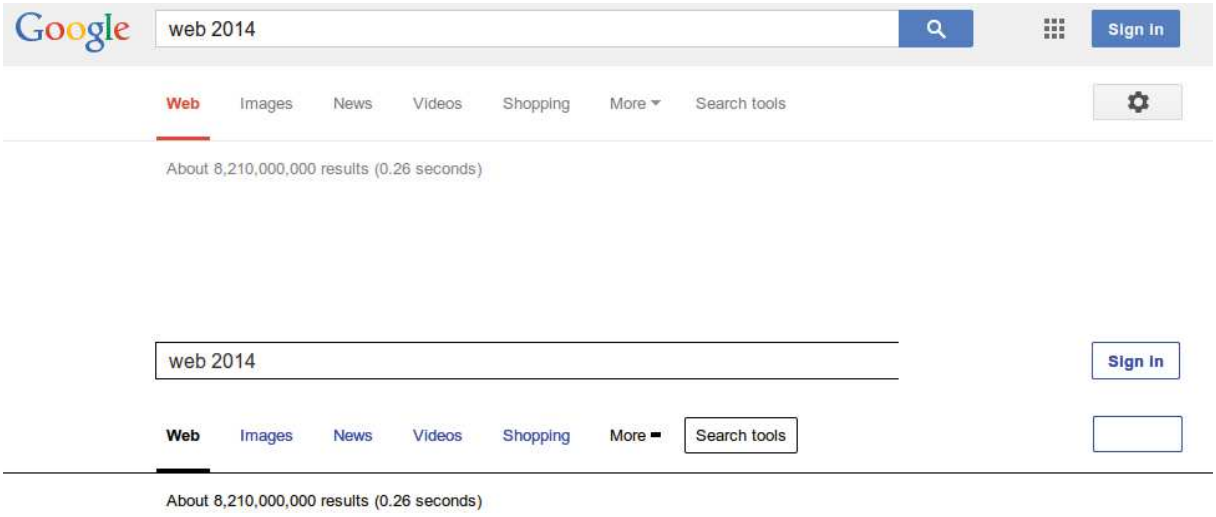


FIGURE 4.10 – Perte d'éléments d'interaction par la désactivation des images d'arrière-plan

La page de résultats du moteur de recherche Google [7] est aussi un très bon exemple de l'impact de la suppression des images d'arrière-plan. Sur cette page de résultats, des éléments d'interaction sont affichés par le biais d'images d'arrière-plan. À droite de la barre de recherche, le bouton pour lancer la recherche a disparu. Même si cette recherche peut être lancée par l'appui sur la touche « entrée » du clavier lorsque l'on est dans le champ

6. http ://www.bbc.com/weather/ (Avril 2014)
7. http ://www.google.com

de recherche, un élément d'interaction a bien été perdu. Il en va de même pour le bouton permettant d'accéder à la liste des applications mais aussi pour le bouton d'accès à la configuration. Ces boutons insérés sous la forme d'images disparaissent complètement de l'affichage enlevant par la même occasion des possibilités d'interaction pour les utilisateurs utilisant les options d'accessibilité du navigateur. C'est le cas sur cet exemple, mais ce genre de problème arrive sur de nombreux sites Web jusqu'à même faire disparaître le logo de la page, rendant parfois la reconnaissance du site difficile.

4.2.4 Ordre du contenu

Des outils permettent de modifier dynamiquement le contenu ou l'organisation d'une page Web (cf. Chapitre 3). Ces outils permettent notamment d'isoler une partie de la page afin de limiter la surcharge d'information. Sur ce point, le masquage des informations qui ne sont pas nécessaires sur le moment apporte un certain bénéfice sur le confort de lecture. Il y a des limites à ces outils mais elles sont bien moins évidentes que pour les autres. En effet, la qualité de la modification est difficile à juger. Ce que l'on constate par contre est qu'il est souvent possible de détecter et d'isoler le contenu principal de la page (par exemple avec l'outil « readability »). Mais il est plus complexe de détecter le menu, l'entête, le pied de page et autres éléments de la page. Cette difficulté provient entre autres du fait que le format de page utilisé pour le développement ne contient pas suffisamment d'informations. Avec HTML5 et ARIA, on peut imaginer que des outils plus poussés soient proposés et profitent de la sémantique importante apportée par ces deux normes pour offrir une meilleure manipulation de la page.

4.2.5 Le daltonisme

De nombreux travaux de recherche tentent de résoudre les difficultés liées aux différentes formes de daltonisme [MAMS14]. Plusieurs de ces approches tentent d'appliquer des thèmes de couleurs adaptés à chaque type de daltonisme sur les pages Web. Il existe donc une correspondance entre un type de daltonisme (ou une variante) et une adaptation. Dans l'article [FS09], les thèmes de couleurs qui sont appliqués sont donc figés (construits une fois pour toutes selon le type de daltonisme). Les avantages de cette méthode sont sa rapidité et sa simplicité. Il suffit de substituer une couleur par une autre couleur. L'inconvénient réside dans le fait que l'on ne tient pas vraiment compte du contexte de couleur d'origine de la page, mais aussi que les thèmes de couleurs ne sont pas vraiment personnalisés et ne peuvent évoluer dans le temps.

4.3 Conclusion

Dans l'ensemble, plus la déficience visuelle est importante et plus les difficultés rencontrées seront nombreuses. La manière dont les pages sont développées y contribue fortement. Dans les standards, des recommandations sont proposées pour permettre au développeur d'assurer une accessibilité minimale sur les sites Web. Malheureusement, les problèmes d'accessibilité sont nombreux sur les pages Web. Les divers outils d'assistance permettent

dans une certaine mesure de combler un certain nombre de problèmes, mais il ne sont pas toujours très efficaces selon les difficultés rencontrées. Nous avons vu dans ce chapitre les limites de ces outils d'assistance et mis en avant leur origine. Le chapitre suivant se concentre sur les méthodes permettant d'obtenir les préférences personnelles d'un utilisateur en terme d'adaptation. Ces préférences seront à la base d'une approche, basée sur la satisfaction de ces préférences, développée durant nos travaux de recherche.

Chapitre 5

Analyse des besoins en vue de l'acquisition des préférences de l'utilisateur

Préambule

Ce chapitre se consacre à l'étude des besoins des utilisateurs déficients visuels ayant une basse vision. De manière à avoir un aperçu des difficultés (desquelles découlent les besoins) des personnes ayant une basse vision, nous avons posé une question assez générale à quelques personnes malvoyantes et avons recueilli leurs réponses. Il s'agit de laisser s'exprimer les personnes sur les éléments qu'elles aimeraient changer sur les pages Web (en général) pour avoir un accès de meilleure qualité à l'information. Les réponses permettront d'avoir une idée des points sur lesquels des transformations devront être effectuées. Au-delà de la mise en évidence du type de préférences que les utilisateurs pourraient exprimer, ce chapitre propose quelques pistes sur les moyens d'acquisition de ces préférences, ou sur la manière de faire exprimer aux utilisateurs leurs préférences. Puisque cette phase d'acquisition n'a pas encore été réalisée, nous proposons deux discussions sur deux techniques différentes permettant de réaliser cette acquisition.

Sommaire

5.1 Introduction **77**
5.2 Types de besoins et leur expression **78**
5.3 Analyse des réponses **79**
5.4 Synthèse des réponses et observations **81**
5.5 Souhaits et préférences **82**
5.5.1 Acquisition par l'exemple 83
5.5.2 Acquisition par apprentissage 84
5.6 Conclusion **85**

5.1 Introduction

Le fondement de ces travaux de recherche repose sur la considération des besoins réels et personnels de chacun. Du fait de sa culture, de son vécu ou de sa santé, toute personne a ses préférences et habitudes qui lui sont propres. Alors que pour beaucoup de personnes ces préférences sont souvent de l'ordre du confort, pour certaines elles peuvent devenir des besoins lorsqu'il n'est pas possible de s'en passer. C'est par exemple le cas des personnes ayant une déficience. Une pathologie quelconque vient bouleverser un ou plusieurs sens nous permettant, entre autres choses, d'acquérir de l'information. Dans le cas de la déficience visuelle, on distingue les personnes ayant un reste visuel suffisant pour accéder encore à de l'information visuelle de celles qui n'en ont plus.

La compensation d'une déficience est un facteur d'importance dans l'expression des besoins par l'utilisateur. En effet, chaque individu a un potentiel d'adaptation différent. Deux personnes ayant une même pathologie vont développer des méthodes différentes afin de compenser leur handicap. Ces méthodes peuvent être d'ordre « externe », par le biais d'outils d'assistance, ou bien « biologique ». Par exemple, lorsque la rétine est endommagée, la personne déficiente visuelle va automatiquement chercher une portion de cette dernière qui est encore suffisamment fonctionnelle et l'utiliser davantage. La vision centrale n'est donc plus forcément le moyen principal de percevoir le monde extérieur avec précision. Des réflexes se développent afin de recalibrer la position des yeux en fonction de ce que l'on cherche à percevoir ou voir. Certaines personnes déficientes visuelles ne regardent pas les personnes dans les yeux lorsqu'elles leur parlent. C'est en tout cas ce que l'on perçoit de l'extérieur, alors qu'en réalité, elles regardent bien la personne à qui elle s'adresse. Si le regard dévie sur le côté c'est simplement parce que la zone qui leur permet de percevoir leur interlocuteur n'est plus la zone centrale. Dans ce type de pathologie, la dégénérescence de la rétine est particulière à chaque personne, la ou les zone(s) « mortes » se répartissent différemment. De ce fait l'adaptation est elle aussi unique.

L'utilisation de l'outil informatique dresse un nombre assez important de barrières à l'accès à l'information pour les personnes ayant une déficience visuelle et les adaptations de chaque personne sont très variées. Le pointeur de la souris qui est un élément mobile peut facilement être perdu lorsque l'utilisateur a une vision parcellaire (avec des trous). Dans ce cas il est par exemple possible de bouger le pointeur jusqu'à ce qu'il entre à nouveau dans une zone de vision alors que les yeux de l'utilisateur sont fixes. Il est également possible de se servir des angles de l'écran comme points de référence. Etant donné que le pointeur ne peut sortir de l'écran, il est très facile de l'amener dans un coin pour le retrouver. Une autre solution pourrait être de garder le pointeur immobile et de balayer l'écran du regard pour le retrouver. De manière générale toutes les techniques de compensation pour un même problème ne conviennent pas à tout le monde. Cela engendre par conséquent des besoins spécifiques lorsque les méthodes de compensation doivent être appuyées par des outils externes. Si on doit effectuer un grossissement du texte alors le taux de grossissement est propre à chacun. Il est évident qu'une personne ayant une vision tubulaire (vision centrale uniquement) sera gênée par un texte de trop grande taille. Elle devra en plus du balayage horizontal de la lecture, balayer verticalement afin de voir l'intégralité des caractères.

Dans la section suivante (Section 5.2) un aperçu des retours que certains utilisateurs ayant une basse vision ont faits est étudié. Ces retours serviront de base et d'exemples

pour l'expression des besoins et préférences de l'utilisateur dans les sections suivantes.

5.2 Types de besoins et leur expression

Les personnes ayant une déficience visuelle connaissent en général leur pathologie et ses effets, cependant elles ont bien plus de mal à exprimer de manière évidente les adaptations réflexes ou non (méthodes de compensation) qu'elles utilisent, parce que la plupart d'entre elles se font de manière inconsciente. De plus, il est aussi difficile d'exprimer ses besoins et surtout de les quantifier. Par exemple une personne qui a besoin d'un certain contraste pour lire un texte par rapport à son arrière-plan ne sera pas spécialement en mesure de donner la valeur minimale du contraste à partir de laquelle un texte devient lisible pour elle. Et pourtant ces valeurs sont importantes pour réaliser une adaptation de qualité.

Afin d'avoir un aperçu plus concret des besoins que peuvent avoir des personnes ayant une basse vision, nous avons posé à un échantillon non représentatif de personnes ayant une basse vision une question assez générale et ouverte. Nous leur avons demandé de formuler ce qu'elles voudraient changer sur les pages Web en général pour y avoir un accès de meilleure qualité. Voici trois des réponses que nous avons pu recueillir.

Réponse 1 : Lorsqu'il m'arrive de naviguer sur des sites difficilement accessibles pour moi, dans la majorité des cas mes difficultés proviennent des points suivants : le site internet donne de nombreuses informations agencées sur la page de façon hasardeuse et avec le zoom il est difficile d'accéder à toutes ces dernières sans devoir repasser plusieurs fois aux mêmes endroits et parfois même, certaines d'entre elles ne nous parviennent pas. Par exemple, lorsque les informations sont placées en colonne et quand il y a des titres en gros caractères, avec le zoom cela est compliqué car il est réglé en fonction de la taille du texte et les titres deviennent très grands et difficiles à lire. Autre problème, lorsque le contraste entre le texte et le fond n'est pas assez élevé, avec ou sans l'inversion de couleur, la lecture est très dure. En outre, lorsqu'une partie de la page n'a pas les mêmes couleurs qu'une autre, il faut parfois jongler entre l'activation et la désactivation de l'inversion de couleur et cela est fastidieux, par exemple si une partie de la page est écrite en blanc sur fond noir et une autre en noir sur fond blanc. Enfin, lorsque les liens hypertextes sont sur des images ou bien lorsque le texte est sous forme d'image, dans certains cas, cela peut devenir gênant.

Réponse 2 : Pour améliorer ma navigation Internet et pouvoir accéder plus aisément sur certains sites difficilement accessibles, j'aimerais bien que l'on puisse changer les contrastes et les couleurs de l'apparence du site, ou qu'il soit tout simplement bien contrasté à la base. De plus, certaines mises en forme des textes et des images sont parfois complexes et un peu éparpillées sur l'ensemble de la page, alors qu'une mise en forme linéaire et claire serait bien plus confortable. Pour finir, ce qui est le plus énervant sur Internet, c'est les sites comportant des publicités ou des éléments dynamiques qui empêchent la lecture par exemple d'un texte qui défilerait ou le visionnage d'une photo ou d'une image qui bougerait ou clignoterait.

Réponse 3 : Voilà, lorsque je navigue sur les sites, ce qui me décourage de continuer sur un site c'est :

- la police de caractères qui n'est parfois pas assez simple ou plutôt trop enjolivée pour pouvoir lire facilement sans forcer.
- lorsque le texte est sur un fond d'écran et que du fait il devient illisible.
- toujours sur la lisibilité, il est parfois difficile, voire impossible, de lire par exemple "valider" ou « continuer » car ces mots sont en couleur dans un encadré qui, lui aussi, est en couleur et, même en utilisant les changements de contraste, cela ne se lit pas du premier coup.
- Il y a aussi des sites qui « envoient » des pub (ex : - 60% sur...) qui viennent se mettre sur ce que l'on est en train de lire et cela est fatiguant car on perd le fil de la lecture.

En résumé, il faudrait épurer les sites avec des caractères simples, un fond d'écran uniforme, des contrastes bien appropriés, et, si l'on veut personnaliser on peut enjoliver avec des images ou autre sur le contour, ce qui laisserait de la place au texte.

Ces trois réponses, bien que non représentatives de l'ensemble des personnes en situation de handicap visuel, sont un élément essentiel pour comprendre et déduire des besoins réels. Les personnes interrogées possèdent chacune une pathologie visuelle différente des autres ce qui permet d'avoir un ensemble de « difficultés » plus large et de coller au mieux à la réalité. De plus, les trois réponses vont pouvoir être comparées pour déterminer les points communs et les points de variation.

5.3 Analyse des réponses

Cette section présente une brève analyse des réponses données par des utilisateurs (cf. Section 5.2). La première étape de l'analyse consiste à extraire, de chacune des réponses données par les personnes interrogées, les idées clés en se détachant de la formulation (langage). De ces idées, on pourra ensuite extraire les difficultés rencontrées qui ont été exprimées. L'objectif étant de n'avoir que les informations nécessaires pour pouvoir les classifier.

La première réponse contient cinq idées importantes menant à des difficultés :

- Quantité d'informations trop importante.
- Agencement de l'information hasardeux (accès aux informations en plusieurs fois, ou inexistant (risque important de passer à côté)).
- Difficulté à adapter le zoom à l'ensemble des tailles de police existantes sur la page.
- Jongler avec plusieurs types de filtres de couleur.
- Liens au-dessus d'images et texte au format image.

La densité d'information est le premier point relevé ici. La quantité d'informations, bien souvent de natures diverses, présentes sur une page peut amener à des difficultés de repérage ou afin de trouver l'information désirée. L'agencement (disposition/organisation) de l'information sur la page semble également donner lieu à des difficultés. Il est vrai que la segmentation de l'information et son organisation sur une page n'est pas toujours très intuitive et pertinente notamment lorsque l'on se sert d'outils d'assistance qui ne permettent pas d'avoir une vue d'ensemble sur la page consultée. Nous avons vu au début de ce chapitre que le taux de grossissement de la police était propre à chacun et qu'une taille trop petite ou trop grande pouvait être gênante. Une difficulté est exprimée sur le zoom (grossissement) par rapport aux tailles de polices différentes sur la page. En effet, si la page consultée contient des tailles de polices très différentes selon le type de texte (titre, paragraphe, etc.) alors il peut être très difficile de trouver un taux d'agrandissement suffisant pour lire les textes écrits en police de plus petite taille et pas trop important pour parvenir à lire les textes écrits en police de plus grande taille. Le quatrième point exprimé concerne une manipulation intensive des filtres de couleur des logiciels d'agrandissement d'écran (magnifiers d'écran). Si la personne a besoin d'avoir un arrière-plan sombre pour lire du contenu textuel alors, si la page Web possède à la fois des zones avec un arrière-plan sombre et des zones avec un arrière-plan lumineux, il n'est pas possible en appliquant un filtre de couleur sur toute la page d'avoir des fonds sombres partout (cf. Chapitre 4). Le dernier point fait référence aux liens de navigation qui peuvent être placés sur un texte se trouvant au-dessus d'une image placée en arrière-plan, mais aussi aux textes au format image. Ici deux idées différentes en ressortent. Les liens textuels ayant des images en arrière-plan peuvent être parfois dissimulés rendant leur détection plus difficile. Concernant les images contenant du texte, il s'agit d'un gros problème pour les personnes ayant un lecteur d'écran, mais aussi pour les personnes qui utilisent une loupe car le grossissement d'un texte au format image donne un rendu qui n'est pas propre. Lorsqu'il s'agit d'un véritable texte le lissage de police permet d'avoir un bon résultat.

La deuxième réponse est plus sommaire mais fait apparaître de nouveaux éléments :

— Changer contraste et couleurs d'apparence de la page.

— Disposition et mise en forme du texte étalées.

— Publicités et éléments dynamiques (flash, clignotement).

Le premier point est un peu plus technique puisque la personne parle de contrastes et de couleurs. Le contraste énoncé ici fait principalement référence au contraste entre le texte et son arrière-plan. Les couleurs d'apparence font, elles, référence au thème visuel de la page. Le point suivant rejoint en partie celui de la réponse précédente sur l'agencement. La disposition du texte et le fait d'avoir un texte sur la page qui est étalé engendre un balayage plus important pour parcourir l'intégralité du contenu textuel. Le dernier point est en quelque sorte tout ce qui concerne la distraction, tout ce qui peut détourner l'attention de la personne de l'information qu'elle cherche à obtenir. Les publicités ajoutant du contenu « inutile » et les animations et flash peuvent tout simplement empêcher la lecture par exemple par éblouissement.

La dernière réponse rejoint sur plusieurs points les précédentes :

— Fioritures sur les polices de caractères.

— Texte sur un fond d'écran (sur une image).

— Difficultés en présence de textes colorés sur fond coloré.

— Perte du fil de la lecture quand il y a des publicités.

Les fioritures sur les polices de caractères peuvent réduire la lisibilité. Il peut effectivement être préféré un texte avec une police de caractères plus régulière. Pour la présence de texte sur des images on en revient un peu à ce qui a été exprimé dans la réponse précédente. Il en va de même pour les textes colorés sur des fonds colorés. Il s'agit ici d'un problème de contraste comme exprimé dans la réponse précédente mais de manière moins technique. Enfin, on a une information de perte du fil de la lecture lorsqu'il y a des publicités insérées (ou autres éléments hors contexte).

5.4 Synthèse des réponses et observations

Les trois réponses étudiées proviennent de trois personnes ayant une pathologie visuelle différente. Néanmoins on constate qu'un certain nombre de difficultés sont communes entre les personnes. Même si elle est exprimée de manière différente la notion de contraste entre un texte et son arrière-plan apparaît plusieurs fois. Cependant il faut bien faire la distinction entre des idées communes exprimées dans le langage naturel et les besoins concrets de la personne. Même si deux personnes font référence au contraste, le type ou le niveau de contraste désiré n'est pas forcément le même. Beaucoup de personnes font référence à la taille du texte, mais le critère de taille est bien différent d'une personne à l'autre. En plus de cela, les besoins d'une même personne peuvent être affectés et varier en fonction du contexte. Le taux de contraste minimum a de fortes chances de ne pas être le même que l'on se trouve à l'intérieur ou à l'extérieur avec une forte luminosité. Cette étude par le biais des retours des personnes permet de montrer une certaine diversité des besoins des personnes, mais surtout qu'il est véritablement complexe de quantifier chacun des besoins d'une personne qui de plus peuvent varier d'une situation à l'autre. Un échantillon plus grand de personnes aurait simplement permis de mettre en avant de nouvelles difficultés rencontrées mais pas de quantifier précisément les besoins.

L'expression et la quantification des besoins est donc essentielle pour appliquer une adaptation cohérente. Il est donc nécessaire de formaliser davantage les besoins de l'utilisateur.

5.5 Souhaits et préférences

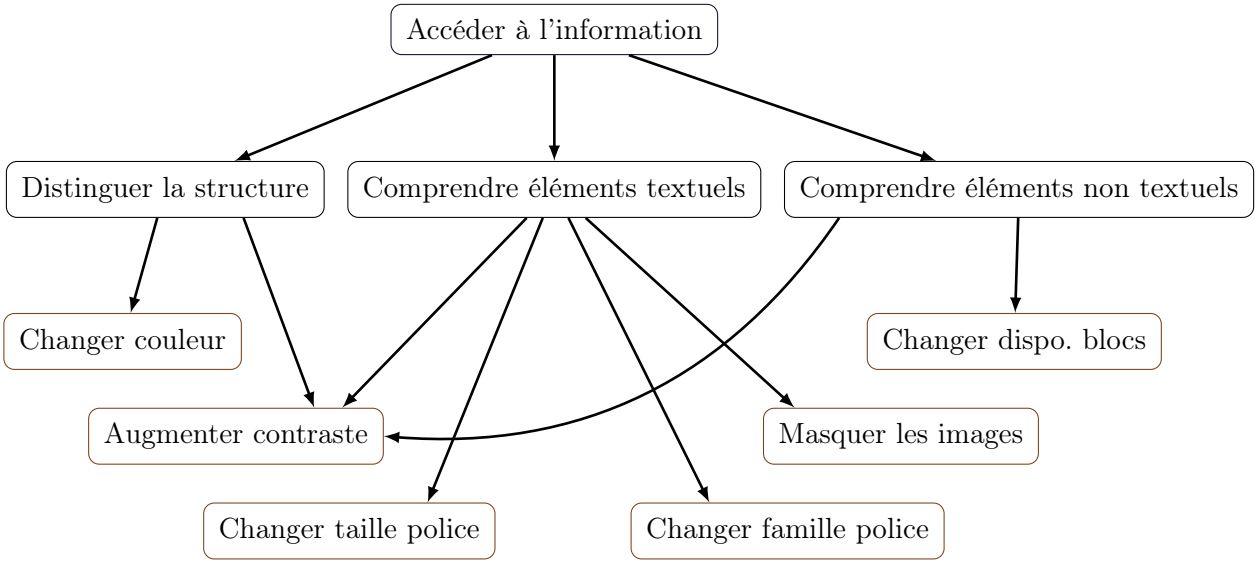


FIGURE 5.1 – Du besoin d'accès à l'information aux souhaits

D'un besoin général d'accès ou d'acquisition de l'information naissent un certain nombre de besoins plus spécifiques, plus précis. Pour accéder à l'information il faut entre autres distinguer la structure des divers constituants de la page, mais aussi comprendre ce qui s'y trouve. La figure 5.1 montre une façon de décomposer les besoins et une évolution possible vers une série de souhaits. Les souhaits correspondent aux préférences de l'utilisateur mais sont décrits dans le langage naturel. La distinction de la structure passe par la perception de formes, de couleurs etc. Il est donc nécessaire que les éléments soient suffisamment distinguables. Pour cela l'utilisateur peut par exemple émettre des souhaits concernant l'augmentation du contraste si ce dernier est insuffisant pour lui. Il peut également demander à avoir une autre couleur si la couleur utilisée lui pose problème comme ce pourrait être le cas si la personne est atteinte de daltonisme. La compréhension se trouve à un niveau un peu au-dessus dans le sens où il ne s'agit plus simplement de distinguer (apercevoir) des structures mais dans ce cas de les voir bien plus en détail. On peut dissocier deux types d'éléments, ceux qui véhiculent de l'information textuelle et ceux qui apportent de l'information par des schémas, images ou illustrations. Concernant le texte il est bien entendu nécessaire d'avoir un contraste suffisant pour qu'il soit lisible. L'utilisateur pourra donc exprimer un souhait de contraste, ce dernier étant d'une certaine manière un peu différent du souhait de contraste qui peut être donné pour la distinction de la structure puisqu'il sera très certainement plus élevé. Le niveau de contraste nécessaire pour lire un texte est plus important que le niveau de contraste nécessaire pour l'apercevoir. La lisibilité d'un texte ne se limite pas uniquement au contraste entre ce dernier et son arrière-plan. La taille du texte et la famille de polices utilisée entrent aussi en ligne de compte. Des souhaits pourraient donc aussi être donnés sur ces propriétés. La figure 5.1 donne quelques exemples, mais elle n'a pas vocation à être prise comme référence. Chaque utilisateur a des besoins et des souhaits différents qui lui sont propres.

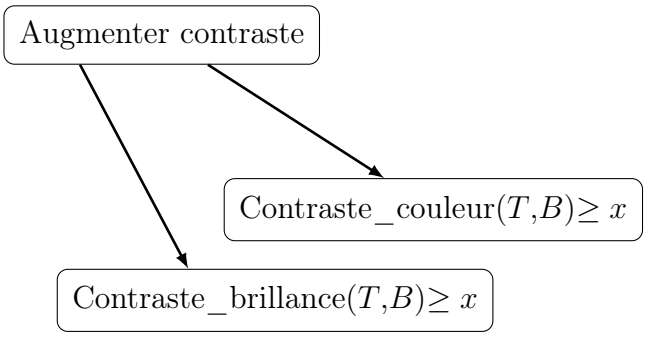


FIGURE 5.2 – Exemple de souhait vers préférence

La formalisation est donc une étape essentielle pour que ces souhaits puissent être manipulés de manière automatique par une machine. On appelle préférence la formalisation d'un souhait. Par exemple, « avoir une taille de police de caractères suffisante » pourra se traduire par $T_c \geq x$ où T_c est la taille de police de caractères et où x est la valeur minimale à avoir. La figure 5.2 ci-contre montre un exemple de passage d'un souhait d'augmentation de contraste (contraste d'origine insuffisant) vers une préférence de contraste minimum. Dans cet exemple on peut voir que le souhait peut se traduire par différentes formes de préférences, par exemple avec un contraste de brillance (le plus courant) ou bien par un contraste de couleur. Pour une description formelle des préférences de l'utilisateur vous pouvez vous reporter au chapitre 6. Ce chapitre décrit un formalisme de préférences qui sera utilisé dans les diverses approches étudiées. La figure 5.2 donne un schéma de réflexion sur le passage du besoin d'acquisition de l'information aux souhaits de l'utilisateur et finalement aux préférences. Dans cet exemple nous donnons quelques idées, évidemment d'autres chemins existent.

Les sections suivantes donnent des pistes pour l'acquisition des préférences de l'utilisateur. En effet, il est loin d'être évident pour un utilisateur d'exprimer lui-même ses préférences de plus cette étape serait très fastidieuse. Il est donc nécessaire d'utiliser des méthodes plus transparentes afin de récupérer ces préférences. L'acquisition par l'exemple et l'apprentissage de préférences sont deux approches différentes permettant à un utilisateur de donner ses préférences.

5.5.1 Acquisition par l'exemple

Pour être plus précis, il s'agit plutôt d'élicitation des préférences de l'utilisateur. L'élicitation consiste à déduire et lister les préférences, dans notre cas, celles d'un utilisateur, sans pour autant qu'il ait à les exprimer une par une. Il s'agit de rendre transparente une étape fastidieuse pour l'utilisateur. L'utilisation d'exemples permet d'avoir des éléments concrets sur lesquels l'utilisateur peut interagir.

Dans le cadre de la transformation de page Web (partie visuelle) l'élicitation des préférences par l'exemple est une approche tout à fait adaptée. Il pourrait s'agir d'utiliser des exemples d'adaptation et de voir si ces adaptations conviennent ou non. Puisque l'on cherche à transformer des pages Web pour des personnes ayant une déficience visuelle, une difficulté se pose quant à l'accessibilité elle-même de l'interface de configuration. En effet, cette dernière se doit d'être complètement accessible car c'est cette interface qui permettra, par le biais de la définition des préférences, d'adapter toutes les pages Web. À ce stade des travaux de recherche, l'interface d'acquisition des préférences n'a pas encore été réalisée. Cependant, nous nous orienterons vers une interface sous la forme d'une page Web. Cette dernière pouvant respecter les différents standards du W3C et être facilement

accessible par divers lecteurs d'écran.

5.5.2 Acquisition par apprentissage

L'acquisition par apprentissage est une autre approche permettant d'obtenir les préférences de l'utilisateur sans qu'il ait à les expliciter une par une en détail. Divers algorithmes peuvent être utilisés pour cet apprentissage. L'apprentissage de préférences ne se limite pas à un seul domaine. De plus en plus l'évolution des technologies et techniques tend à rendre la plus transparente possible cette phase d'acquisition des préférences [RDK08].

L'algorithme Q-learning est un algorithme d'apprentissage par renforcement qui se base sur le principe de bonus/malus. Cet algorithme a déjà été utilisé dans des travaux de recherche proches portant sur l'apprentissage des préférences utilisateurs pour la personnalisation de pages Web [FMPS14a]. Dans cette utilisation du Q-learning, un certain nombre de modifications peuvent être réalisées sur les différents éléments de la page. Lorsque l'utilisateur accède à une page, il peut modifier certains éléments. Lorsque l'utilisateur modifie un élément alors un processus se lance afin d'apprendre de ces modifications. Un élément particulier qui est toujours transformé de la même manière se verra augmenter son compteur. Si ce dernier pour une raison ou une autre est modifié d'une manière différente alors un malus sera appliqué. Dans les travaux de recherche de S. Mirri et al. [FMPS14a], l'algorithme de Q-learning a été utilisé afin de générer des profils utilisateur. Les profils utilisateur sont ensuite utilisés sans altération, c'est-à-dire qu'il n'y a plus de prise en compte des modifications potentielles des préférences.

Cet algorithme d'apprentissage par renforcement permet donc de déduire un certain nombre de préférences de l'utilisateur. Cependant, à la différence de l'utilisation décrite précédemment, l'algorithme d'apprentissage doit être en fonctionnement quasi permanent (toujours prêt à recevoir de nouvelles informations). En effet, les préférences de l'utilisateur peuvent à tout moment évoluer. D'autres algorithmes, qu'ils soient dérivés ou non du Q-learning, pourraient être adaptés à notre contexte, mais ce qui fait son intérêt c'est l'interface utilisateur qui permet de le manipuler. Bien sûr l'utilisateur ne doit pas directement avoir conscience qu'il utilise un algorithme d'apprentissage, mais l'interface qu'on lui propose doit lui permettre d'appliquer les modifications qu'il souhaite réaliser et ces modifications doivent d'une manière ou d'une autre être conservées en mémoire, de manière à ce que la modification d'une page Web ait un impact sur les prochaines pages visitées.

L'interface permettant de réaliser des modifications sur une page donnée doit être en mesure d'offrir un grand potentiel de modifications. Jusqu'ici, les transformations possibles sont des transformations que l'on pourrait qualifier de simples dans le sens où elle s'appliquent à un élément en particulier et uniquement à celui-ci. Par exemple on peut, pour un niveau de titre donné, sélectionner une couleur. Chaque fois que l'on va changer la couleur de ce niveau de titre, le système d'apprentissage permettra de déduire la préférence de l'utilisateur concernant la couleur de titre qu'il souhaite avoir. Il est également possible de faire ceci sur des tailles de polices, des familles de police ou autres. Cependant, il n'existe pas vraiment d'interfaces permettant de donner des règles entre plusieurs éléments. Par exemple indiquer un contraste entre un texte et son arrière plan. Ou bien, donner une différence de taille minimale entre un contenu textuel (paragraphe) et son titre. Il est donc nécessaire de concevoir une interface permettant à l'utilisateur d'exprimer ce genre

de préférences.

Ces deux dernières sections présentent deux approches différentes permettant d'acquérir des préférences de l'utilisateur. Cependant, il n'est pas nécessaire que ces deux approches soient déconnectées. Un processus d'apprentissage (intégrant un algorithme d'apprentissage) pourrait aussi être utilisé avec une interface basée sur des exemples. L'acquisition par l'exemple ayant l'avantage de donner une interface relativement concrète pour l'utilisateur. D'autant plus que les préférences qui pourront être définies peuvent porter sur des éléments de haut niveau. Ceci ouvrant donc un autre champ de recherche portant sur la détection et la reconnaissances des blocs visuels d'une page Web. Le chapitre suivant traite à la fois de cette détection et de cette identification, mais aussi des méthodes de manipulation à proprement parler des éléments des pages.

5.6 Conclusion

Le choix de la méthode utilisée pour l'acquisition des préférences est très important. En effet, les préférences de l'utilisateur peuvent difficilement être explicitement données une à une par l'utilisateur. Une méthode doit donc être utilisée afin de rendre cette étape la plus simple et transparente possible. L'acquisition des préférences par l'exemple et l'apprentissage de préférences sont deux méthodes différentes mais néanmoins efficaces pour réaliser cette étape. La combinaison des deux méthodes peut tout à fait être envisagée afin d'obtenir une approche intermédiaire.

Une fois que nous avons l'ensemble de préférences de l'utilisateur, il faut être en mesure de savoir exactement sur quels éléments de la page elles s'appliquent mais surtout d'avoir ces éléments ainsi que leurs différentes propriétés. Par exemple lorsqu'une préférence de taille de police de caractères minimale est définie, alors il faut être en mesure de trouver tous les textes sur la page et de connaître leur taille.

Chapitre 6

Recherche d'une adaptation

Préambule

Dans ce chapitre, nous présentons les différentes pistes que nous avons envisagées pour résoudre le problème du calcul « rapide » d'une adaptation. Nous présenterons tout d'abord une approche exacte qui se base sur la théorie des préférences (section 6.1). Les limites de cette approche pour la résolution de notre problème sont très vite apparues. Cependant cette approche pose les fondamentaux de la formalisation des préférences. Dans les sections 6.2 et 6.3 nous proposons un langage d'expression des préférences que nous souhaitons traiter. Puis la section 6.4 présente de manière plus détaillée une approche basée sur deux variantes de la méta-heuristique NSGA (Non-dominated Sorting Genetic Algorithm). Les deux variantes utilisées sont des algorithmes évolutionnaires permettant notamment de résoudre des problèmes d'optimisation. Nous présentons une partie de la mise en œuvre relative aux opérateurs génétiques et aux fonctions objectif.

Sommaire

6.1 Approche exacte - La théorie des préférences **87**
6.2 Syntaxe du langage des préférences **88**
6.3 Sémantique du langage des préférences **89**
6.3.1 Méthodes de résolution 90
6.3.2 Limites principales liées au contexte 92
6.4 Les algorithmes évolutionaires **93**
6.4.1 L'algorithme NSGA - Concepts fondamentaux 94
6.4.2 La variante NSGA-II 96
6.4.3 La variante NSGA-III 97
6.4.4 Le tri par fronts de non domination 98
6.4.5 Les opérateurs génétiques 100
6.4.6 Les fonctions objectif 108
6.5 Conclusion **113**

6.1 Approche exacte - La théorie des préférences

La théorie des préférences [Kac11] est un domaine regroupant modélisation et méthodes de résolution de problèmes liés aux préférences. Dans notre cas, il s'agit de manipuler des préférences utilisateurs concernant la visualisation de pages web. Les algorithmes de résolution permettent entre autres de gérer les dépendances potentielles entre les préférences définies par l'utilisateur et ainsi d'ordonner les différentes solutions.

Dans le domaine de la théorie des préférences, la représentation des préférences n'est pas unique. Il existe plusieurs façons de donner une description formelle d'un même problème [BBB13, BBHP99, HH92]. La plupart des approches reposent sur des préférences simples comme « Le Rouge est préféré au Bleu pour le texte » ou sur des préférences conditionnelles comme « Si la couleur de police est Verte alors le style Gras est préféré au style Régulier ».

Pour modéliser notre problème dans cette approche, nous introduisons un ensemble de variables $V = \{V_1, V_2, \ldots, V_n\}$ qui représente l'ensemble des objets (HTML) utiles de notre page associés à leurs caractéristiques. Plus précisément, une variable représente l'association d'un élément HTML avec l'une de ses caractéristiques. Par exemple, la variable « Taille du titre » associe à l'élément HTML titre la caractéristique de taille. Chaque variable V_i possède son propre domaine, noté $Dom(V_i)$, qui représente l'ensemble des valeurs possibles pour cette variable (les tailles, les couleurs, etc.). **Un tuple issu du produit cartésien des domaines de toutes les variables est appelé une solution**. Par exemple, $V_1 = 3$, $V_2 = 1$, $V_3 = 7$ est une solution associant respectivement la valeur 3 à V_1, 1 à V_2 et 7 à V_3, variables ayant pour domaine de définition $Dom(V_1) = Dom(V_2) = Dom(V_3) = \{1, 2, 3, 4, 5, 6, 7\}$. Les domaines de définition peuvent être de natures diverses. De plus un domaine de définition n'est pas forcément un ensemble ordonné, ni muni d'une fonction de comparaison. Le terme de solution est utilisé ici quel que soit le degré de satisfaction des préférences par cette combinaison de valeurs : une solution peut satisfaire ou non les préférences. La taille de l'espace de recherche (nombre de solutions) s'exprime par la formule suivante :

$$Search\ space\ size = \prod_{i=1}^{n} |Dom(V_i)| \tag{6.1}$$

L'ensemble des solutions est noté $\Omega = \{\omega_1, \omega_2, \ldots, \omega_k\}$ avec k le nombre de solutions dans l'espace de recherche. Dans le but de réduire l'espace de recherche, une phase d'élimination des solutions impossibles (infaisables, incohérentes) peut être effectuée. L'impossibilité peut venir de plusieurs critères. Par exemple, on peut avoir une impossibilité physique ou logique lorsqu'on traite par exemple un problème concret sur du positionnement géographique : un objet ne peut pas être à deux endroits en même temps. L'impossibilité peut également venir d'une contrainte matérielle, conceptuelle ou contextuelle : ce modèle de téléphone n'existe pas en vert. Cependant, selon la nature du problème, si le nombre de solutions impossibles est négligeable par rapport au nombre de solutions totales, il est tout à fait possible de ne pas effectuer cette tâche de suppression. Il suffira ensuite d'ignorer ces solutions dans la relation de préférence finale.

6.2 Syntaxe du langage des préférences

La modélisation des préférences utilise le langage formel $\mathcal{L}$, constitué d'un ensemble de formules logiques utilisant des opérateurs de comparaison ainsi que des opérateurs booléens. La grammaire $G = \{V_T, V_N, R, pref\}$ (avec V_T le vocabulaire terminal, V_N le vocabulaire non terminal, R les règles de production, $pref$ l'axiome de la grammaire) est définie ci-après.

Les symboles terminaux du langage sont donnés par l'ensemble suivant :

$$V_T = V \cup \bigcup_{V_i \in V} Dom(V_i) \cup \{(,), =, <, >, \neq, \leq, \geq\} \cup \{\vee, \wedge, \implies, \neg\}$$

Les symboles non terminaux du langage sont donnés par l'ensemble suivant :

$$V_N = \{pref, atome, formule, opcmp, opbool\}$$

Enfin, les règles de production de R sont :

$$atome ::= V_i \ opcmp \ d \quad \text{(avec } d \in Dom(V_i)\text{)} \tag{6.2}$$

$$::= V_i \ opcmp \ V_j \tag{6.3}$$

$$opcmp ::= < \mid > \mid = \mid \neq \mid \leq \mid \geq \tag{6.4}$$

$$formule ::= atome \tag{6.5}$$

$$::= (formule \ opbool \ atome) \quad \text{(conjonction, disjonction, implication)} \tag{6.6}$$

$$::= (\neg \ formule) \quad \text{(négation logique)} \tag{6.7}$$

$$opbool ::= \vee \mid \wedge \mid \implies \tag{6.8}$$

$$pref ::= formule \succ formule \quad \text{(formule de gauche préférée à celle de droite)} \tag{6.9}$$

Les conventions d'écriture usuelles de la logique des propositions permettront, par la suite, de ne pas écrire certaines parenthèses.

6.3 Sémantique du langage des préférences

Une interprétation d'une formule de $\mathcal{L}$ est une solution de Ω. Une solution satisfaisant la formule p est appelée modèle de p. Une solution falsifiant la formule p est appelée contre-exemple de p. En prenant l'exemple de véhicules colorés de la table 6.1, ω_1 et ω_2 satisfont la formule $V_1 = blue$ et toutes les autres solutions la falsifie. ω_1 falsifie la formule $V_1 = red \vee V_2 = bike$. Avec le même exemple, en considérant la formule $p = (V_1 = red \wedge V_2 = bike)$, ω_4 est l'unique modèle de p.

Table 6.1 – Véhicules colorés

ω_i	color : V_1	vehicle : V_2
ω_1	$blue$	car
ω_2	$blue$	$bike$
ω_3	red	car
ω_4	red	$bike$
ω_5	$green$	car
ω_6	$green$	$bike$

Une préférence du langage $\mathcal{L}$ exprime qu'une formule est préférée à une autre : $p \succ q$. D'après la logique de Von Wright [VW63], la sémantique de cette préférence est exprimée par : $p \wedge \neg q$ *est préféré à* $q \wedge \neg p$.

Si on donne la préférence $V_1 = red \succ V_1 = blue$ (que l'on peut écrire par abus de notation $red \succ blue$) alors les solutions ayant comme valeur red pour V_1 seront préférées aux solutions ayant la valeur $blue$ pour cette même variable. De ce fait, ω_3 $(red - car)$ et ω_4 $(red - bike)$ sont préférées à ω_1 $(blue - car)$ et ω_2 $(blue - bike)$. L'opérateur $\succeq$ indique la relation de préférence entre des ensembles de solutions. On peut donc écrire, pour l'exemple précédent, $\{\omega_3, \omega_4\} \succeq \{\omega_1, \omega_2\}$ pour définir cette relation de préférence. Aucune autre solution n'est concernée par cette préférence et n'apparaît dans cette relation de préférence entre ensembles de solutions. Les deux expressions (6.10) et (6.11) permettent de définir le membre gauche (L) et le membre droit (R) de la relation de préférence pour une préférence $p \succ q$ donnée. On peut imaginer des préférences un peu plus complexes telles que : $V_1 = blue \wedge V_2 = car \succ V_1 = green \wedge V_2 = car$ (les voitures bleues sont préférées aux voitures vertes). Pour cette préférence, la relation de préférences sera $\{\omega_1\} \succeq \{\omega_5\}$. Les préférences peuvent concerner deux variables dont une est en partie gauche et l'autre en partie droite. Par exemple, $V_2 = bike \succ V_1 = red$, on préfère avoir un vélo plutôt que n'importe quel véhicule rouge. Les solutions où l'on a $V_2 = bike \wedge \neg V_1 = red$ sont préférées aux solutions où on a $\neg V_2 = bike \wedge V_1 = red$. Pour cette préférence, la relation de préférences sera $\{\omega_2, \omega_6\} \succ \{\omega_3\}$.

$$L_{p \succ q} = \{\omega_j \in \Omega \ / \ \omega_j \ satisfies \ p \wedge \neg q\} \tag{6.10}$$

$$R_{p \succ q} = \{\omega_j \in \Omega \ / \ \omega_j \ satisfies \ q \wedge \neg p\} \tag{6.11}$$

Lorsque les deux composantes p et q de la préférence portent toutes les deux sur la même variable alors les formules $p \wedge \neg q$ et $q \wedge \neg p$ peuvent-être simplifiées en p et q.

Certaines préférences utilisateur, comme le fait d'avoir une taille minimale, imposent une décomposition en plusieurs préférences de ce formalisme. Par exemple, une préférence telle que « Taille de police d'au minimum 14pt » doit être dupliquée autant de fois que nécessaire afin de couvrir tout le domaine des tailles de polices. Par exemple, si nous disposons de quatre tailles de police différentes qui sont : 10pt, 12pt, 14pt et 16pt alors $Dom((V_1) = \{10, 12, 14, 16\}$ où V_1 est la taille de police. La modélisation de la préférence complexe donnera lieu à quatre préférences plus simples :

— $V_1 = 14 \succ V_1 = 10$
— $V_1 = 14 \succ V_1 = 12$
— $V_1 = 16 \succ V_1 = 10$
— $V_1 = 16 \succ V_1 = 12$

Par conséquent, pour une préférence utilisateur donnée, une ou plusieurs préférences doivent être explicitées dans cette méthode de modélisation. Dans le cas de préférences comme celle concernant une taille minimale, plus le domaine de taille est grand et plus le nombre de préférences dans la modélisation sera important.

Les préférences modélisées sont ensuite stockées dans un ensemble de préférences noté $\mathcal{P}$. L'ensemble de toutes les relations de préférences est noté $\mathcal{L}(\mathcal{P})$ et est aussi parfois appelé ensemble comparatif.

Avec la table précédente (Table 6.1), si on définit les deux préférences simples : $\mathcal{P} = \{red \succ_1 blue, bike \succ_2 car\}$, nous obtenons l'ensemble de relations de préférence suivant :

$$\mathcal{L}(\mathcal{P}) = \{C_1 = (\{\omega_3, \omega_4\}, \{\omega_1, \omega_2\}), \\ C_2 = (\{\omega_2, \omega_4, \omega_6\}, \{\omega_1, \omega_3, \omega_5\})\}$$

Nous précisons que $C_i = (L_i, R_i)$ représente la relation de préférence pour la préférence i.

6.3.1 Méthodes de résolution

La méthode de résolution se base donc sur deux ensembles distincts. Le premier est l'ensemble des préférences utilisateurs modélisées dans le formalisme décrit précédemment et le second est l'ensemble des solutions. En théorie des préférences, il existe deux grandes façons d'aborder la résolution d'un problème. Soit en passant par une méthode explicite où toutes les relations de préférence sont détaillées soit en passant par une méthode dite implicite [Kac11]. Dans la méthode explicite, les relations de préférences contiennent la liste exhaustive de toutes les solutions contenues dans les membres droits et gauches associés aux préférences. Les solutions ainsi listées doivent d'abord être générées (ou construites) afin qu'elles puissent être évaluées et réparties. Cette étape de génération de solutions peut prendre du temps lorsque celles-ci deviennent nombreuses. La méthode de résolution implicite a donc l'avantage de passer outre cette phase de génération des solutions. Elle utilise des formules afin de décrire des groupes de solutions au lieu de les manipuler individuellement. Ainsi la quantité de solutions/groupes manipulés est réduite. Cependant, le bénéfice n'est pas toujours présent. Dans certains cas les préférences amènent à manipuler des formules, décrivant les groupes de solutions, qui deviennent de plus en plus précises. De ce fait, les groupes de solutions peuvent devenir très petits ce qui, à terme, revient quasiment à utiliser une méthode explicite.

Indépendamment du type de méthode, la résolution consiste à organiser l'ensemble des solutions selon la manière dont elles satisfont chacune des préférences définies. Ce tri permet de récupérer les solutions répondant à certains critères (selon l'algorithme de résolution utilisé). À partir de l'ensemble des relations de préférence (chacune associée à

une préférence), différentes sémantiques peuvent être utilisées afin d'ordonner différemment les solutions. Plusieurs sémantiques existent telles « Optimiste », « Pessimiste », « Strong » ou « Ceteris Paribus », cette dernière étant la base des réseaux CP-Net (Ceteris Paribus Network) [Kac11].

En fonction de la sémantique utilisée, les conditions de sélection d'une solution dans les relations de préférence peuvent varier. Mais dans tous les cas, chaque lot de solutions extraites lors d'une sélection constitue une strate (un groupe) de solutions incomparables entre elles (il n'est pas possible de dire qu'une solution est préférée à une autre dans ce groupe). L'algorithme effectuera par conséquent autant d'itérations que nécessaire jusqu'à obtenir un ensemble vide de solutions extraites ou bien jusqu'à ce que l'ensemble de relations de préférence soit vide.

Pour illustrer cette méthode de résolution, nous utilisons la sémantique « Strong » sur un exemple. Cette sémantique a deux variantes : la sémantique « Strong - Minimum of specificity » et la sémantique « Strong - Maximum of specificity ». Ces deux spécificités permettent de choisir l'ordre de sortie des solutions, c'est-à-dire, dans quel sens sera ordonnée la relation de préférence générale (depuis les meilleures solutions aux moins bonnes ou inversement).

Que ce soit l'une ou l'autre des variantes, la sémantique « Strong » fonctionne sur le même principe, seul l'ordre des solutions change. À partir de $\mathcal{L}(\mathcal{P})$ nous cherchons les solutions qui n'apparaissent dans aucune partie droite de relation de préférence. La partie droite contient les solutions pour lesquelles au moins une autre solution est préférée. De ce fait, on choisit donc les solutions qui ne sont pas explicitement considérées comme plus mauvaises qu'une autre. Dans notre exemple précédent, il s'agit de ω_4 et ω_6 qui n'apparaissent dans aucune partie droite. Ces deux solutions ω_4 et ω_6 sont supprimées de la partie gauche des relations de préférence. Nous obtenons ainsi à l'itération 2 un nouvel ensemble de relations de préférence $\mathcal{L}_2(\mathcal{P})$:

$$\begin{aligned}\mathcal{L}_2(\mathcal{P}) = \{&C_1 = (\{\omega_3\}, \{\omega_1, \omega_2\}),\\ &C_2 = (\{\omega_2\}, \{\omega_1, \omega_3, \omega_5\})\}\end{aligned}$$

Comme toutes les solutions restantes apparaissent à présent dans les ensembles de droite, il n'est plus possible d'extraire d'autres solutions. L'exécution de l'algorithme s'arrête ici. Pour les solutions restantes, soit on déclare une inconsistance et on ne tient pas compte de ces solutions, soit il est possible de les considérer comme étant les moins bonnes. Dans ce dernier cas, les solutions seront regroupées dans un ensemble de solutions incomparables à la fin de notre relation de préférence générale. En considérant les solutions restantes voici la relation de préférence générale obtenue pour notre exemple :

$$\succeq = (\{\omega_4, \omega_6\}, \{\omega_1, \omega_2, \omega_3, \omega_5\})$$

Dans cette relation de préférence, la solution $\omega_4 = red - bike$ fait partie du premier groupe de la relation obtenue, ce qui semble évident ($red \succ_1 blue, bike \succ_2 car$). On peut remarquer que la solution $green - bike$ fait également partie du premier groupe de solutions (les meilleures). Cela peut paraître étrange au vu des préférences données au début. Mais avec la sémantique « Strong » et la variante « Minimum of specificity », une solution telle

que $green - bike$ sort en tête puisque nous n'avons pas de connaissance définie pour cette couleur alors que *bike* est préféré à *car*. Si nous avions utilisé la sémantique « Optimiste », le résultat serait le même pour cet exemple. Cependant le fonctionnement de la sémantique « Optimiste » est un peu différent. Lorsque les solutions n'apparaissant pas à droite de l'ensemble des relations de préférences sont choisies, alors toutes les relations de préférence possédant (dans leurs parties gauches) ces solutions sont supprimées. Lorsqu'une relation de préférence est supprimée on dit qu'elle est satisfaite. Dans notre exemple les deux relations de préférences sont supprimées dès la première extraction des solutions (Les solutions extraites apparaissent dans les deux parties gauches) et nous obtenons la même relation de préférence générale.

6.3.2 Limites principales liées au contexte

La théorie des préférence permet de modéliser et d'aborder la résolution de problèmes comme le nôtre (adaptation d'éléments des pages Web), cependant dans le cas pratique, certaines limites apparaissent, notamment liées au fait que la transformation de pages Web engendre la manipulation de nombreux éléments avec leurs caractéristiques associées (représentées par des variables).

Dans la méthode explicite de résolution, la première étape consiste à construire l'ensemble des individus possibles. Toute combinaison de valeurs possibles correspondant à une adaptation potentielle doit être explicitée. Le nombre de solutions à générer dépend donc à la fois du nombre de variables et de la taille du domaine de ces variables. Si nous avons par exemple les deux variables C_T pour la couleur du texte et C_F pour la couleur de fond, alors le nombre de solutions possibles est $|Dom(C_T)| \times |Dom(C_F)|$. Avec simplement 256 couleurs dans le domaine on a 256^2 solutions soit 65536 solutions. Si le nombre de variables atteint une dizaine ce qui est tout à fait cohérent pour des sites Web de petite taille alors le nombre de solutions atteint 256^{10} soit environ 1.2×10^{24} solutions.

L'étape suivante consiste à produire pour chaque préférence la relation de préférence associée, c'est-à-dire que chaque solution doit être évaluée pour chaque préférence. La complexité de ces deux premières étapes, lorsque le domaine de toutes les variables est le même, est dans $O(|Dom(v)|^n \times m)$ avec n le nombre de variables et m le nombre de préférences. Il est donc difficile voire impossible de trouver une bonne solution dans un temps raisonnable. Par temps raisonnable, on entend un temps de quelques secondes seulement. Il nous semble que l'on ne peut aller au-delà de quelques secondes, car ce temps de recherche d'une solution vient s'ajouter au temps de chargement de la page Web et que le processus est lancé à chaque chargement de page. La taille importante de l'espace de recherche empêche donc d'utiliser la méthode de résolution explicite indépendamment de la sémantique choisie.

L'approche par résolution implicite est un peu différente. L'ensemble des solutions construites par combinaison des valeurs des variables n'est pas généré. Lors de la construction des relations de préférences de chaque préférence, les ensembles gauche et droit de solutions sont décrits en intention. La logique propositionnelle est donc utilisée afin de décrire ces ensembles de solutions. Lorsque l'on ajoute ou enlève une solution, une proposition est ajoutée à la formule logique afin de représenter cette inclusion ou exclusion. La résolution se base donc sur ces formules logiques et donne la relation de préférence finale sous la

forme de groupes de solutions décrites par ces mêmes formules. Il reste donc à générer les solutions correspondantes. Une formule logique décrivant un ensemble de solutions pourra être par exemple $c_T = blue \wedge contrast(c_T, c_F) \geq x$ qui peut se lire « Texte blue **et** contraste entre texte et arrière-plan supérieur ou égal à $x\%$ ».

En résolution implicite il n'y a pas de phase de génération de l'ensemble des solutions, ce qui rend le processus bien plus rapide. La complexité pratique de cet algorithme est par conséquent réduite. Mais malheureusement, le problème est simplement déplacé au moment où l'algorithme a fini de s'exécuter et que l'on dispose de la relation de préférence. Les solutions sont décrites à ce moment-là par des formules logiques propositionnelles. Il est donc nécessaire de retrouver les solutions qui correspondent à ces formules. Si on travaille avec des valeurs stockées cette étape est immédiate, mais lorsque l'on utilise des préférences sur des valeurs calculées la recherche de la solution correspondante est plus complexe. Si la formule nous indique que la meilleure solution est celle qui a exactement $x\%$ de contraste entre le texte et l'arrière-plan alors le problème est effectivement en quelque sorte résolu. Mais nous ne connaissons pas les couleurs du texte et de l'arrière-plan. Il faut donc trouver une couleur pour le texte et une couleur pour l'arrière-plan qui donnent un contraste égal à $x\%$. Ceci implique de parcourir une partie de l'espace de recherche et dans le pire des cas l'intégralité des combinaisons de couleurs (si la bonne est la dernière rencontrée).

Les algorithmes issus de la théorie des préférence permettent d'obtenir les meilleures solutions d'adaptation. De plus ils sont déterministes ce qui permet d'avoir la même adaptation à chaque exécution pour une page et un ensemble de préférences donné. Néanmoins il n'est pas possible de trouver de solutions d'adaptation dans un temps raisonnable compte tenu du nombre d'éléments à traiter sur les pages Web. Une expérimentation sur les différentes sémantiques a été réalisée. Elle a permis d'illustrer et de confirmer en pratique les barrières théoriques décrites précédemment. Les résultats sur quelques exemples sont donnés en section 8.2.

6.4 Les algorithmes évolutionaires

Les algorithmes évolutionnaires [CHS15] s'inspirent de la théorie de l'évolution et s'attachent à résoudre des problèmes d'optimisation en faisant évoluer un ensemble de solutions (ou individus) au fil de leur exécution. L'évolution des individus tente de générer de nouveaux individus parmi lesquels se trouveront de meilleures solutions. Ces algorithmes itératifs ont une part d'aléatoire à chaque itération et appartiennent par conséquent aux algorithmes dits stochastiques. Ces algorithmes sont souvent associés à la famille des métaheuristiques mais peuvent également être considérés comme des algorithmes d'intelligence calculatoire (ou artificielle).

Les algorithmes génétiques sont des algorithmes évolutionnaires qui font évoluer une population d'individus par le biais d'un ensemble réduit d'opérateurs génétiques simples. On y trouve par exemple le croisement et la mutation, inspirés d'événements que l'on retrouve dans le domaine de la génétique. Il existe plusieurs types d'algorithmes génétiques. Parmi eux on trouve des algorithmes *simple-objectif* ou des algorithmes *multi-objectifs*. Un objectif est évalué par le biais d'une fonction qui mesure à quel point il est satisfait. Nous appellerons par la suite *fonction objectif* ces fonctions d'évaluation.

Bien souvent les problèmes d'optimisation provenant du monde réel ne se limitent pas à la satisfaction d'un seul et unique but. Ces problèmes sont souvent complexes et l'évaluation des solutions possibles repose sur de nombreux critères qui peuvent être dépendants les uns des autres, voire même conflictuels. La nature du problème d'adaptation de page Web en fait un de ces problèmes complexes. Une approche mono-objectif est donc peu adaptée.

Dans le cas des algorithmes *simple-objectif*, toutes les contraintes que l'on souhaite satisfaire doivent être évaluées par une seule et unique fonction objectif qui peut être définie par agrégation des résultats de plusieurs fonctions objectif évaluant séparément la satisfaction des contraintes. Cette agrégation peut être par exemple une moyenne pondérée si les fonctions sont définies dans une même échelle. Dans le cas des algorithmes génétiques multi-objectifs, chaque contrainte est évaluée séparément des autres par une fonction objectif. Cela permet notamment de trier notre population pour un objectif donné et d'éviter de compenser un mauvais résultat sur une contrainte par un bon résultat sur une autre.

L'algorithme NSGA-II [DAPM02a] (Non-dominated Sorting Genetic Algorithm) a été reconnu en 2009 comme étant certainement le plus populaire [Lie09]. Il s'agit d'un algorithme génétique multi-objectifs. Depuis peu, une nouvelle variante de cet algorithme est parue. Cette troisième version de NSGA vient palier le manque d'efficacité de NSGA-II lorsque le nombre d'objectifs devient important. Ces deux algorithmes (NSGA-II et NSGA-III) ont été étudiés dans notre contexte et une mise en œuvre est proposée. Dans cette mise en œuvre, certains points relèvent de l'interprétation principalement pour NSGA-III [DJ14], puisqu'aucune implémentation officielle n'existe encore et que l'article original [DJ14] passe sous silence certains points particuliers.

Les sections suivantes présentent plus en détail les deux versions (II et III) de NSGA utilisées. En section 6.4.1, les concepts communs aux deux variantes sont mis en avant afin de présenter globalement le fonctionnement de l'algorithme. Les sections 6.4.2 et 6.4.3 détaillent les spécificités propres à chacune des deux variantes. Nous présentons ensuite les opérateurs génétiques tels qu'il ont été définis et utilisés dans la mise en œuvre des algorithmes (Section 6.4.5).

6.4.1 L'algorithme NSGA - Concepts fondamentaux

L'algorithme NSGA a évolué depuis ses débuts afin de dépasser certaines limitations rencontrées. Il existe actuellement trois grandes versions (ou variantes) de NSGA. En dépit de son évolution, un certain nombre de concepts communs restent présents et font la signature de cet algorithme. Cet algorithme n'est pas unique et partage plusieurs concepts communs avec d'autres algorithmes évolutionnaires [Lie09].

NSGA se base sur trois grands types d'opérateurs : le croisement, la mutation et la sélection. Ces opérateurs sont appelés opérateurs génétiques et s'appliquent sur une population. Il permettent de générer une descendance mais aussi de sélectionner les individus qui feront partie de la prochaine génération de la population. Le schéma 6.1 décrit l'utilisation de ces opérateurs génétiques lors d'une itération de l'algorithme (appelée génération).

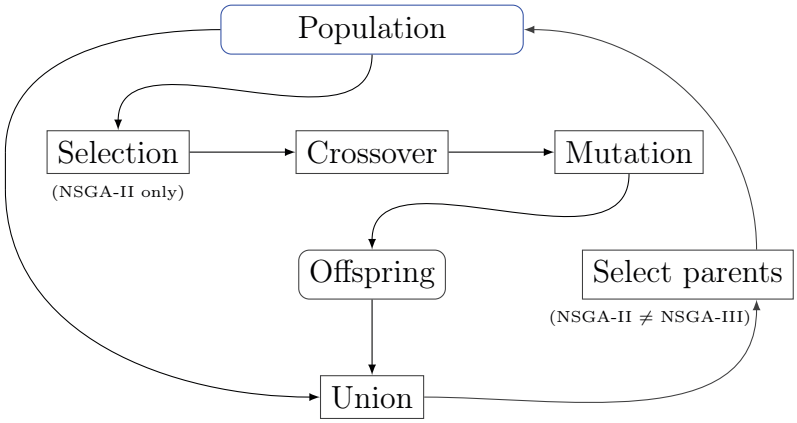


FIGURE 6.1 – Schéma général d'une itération des algorithmes NSGA-II and NSGA-III

Une population est un ensemble d'individus. Un individu est une solution potentielle au problème. Dans notre contexte d'adaptation de page Web, un individu sera une adaptation potentielle de la page (satisfaisante ou non). La population initiale qui constitue le point d'entrée de l'algorithme est initialisée aléatoirement. Pour une taille de population N, on choisit de manière équiprobable (sans biais) N individus dans l'espace de recherche. Du fait d'une sélection aléatoire avec remise (on n'enlève pas l'individu que l'on vient de choisir) il est possible d'avoir plusieurs occurrences d'un même individu dans la population initiale. Les opérateurs génétiques sont identiques dans les deux variantes. Le croisement et la mutation s'appliquent tous deux sur une population et génèrent un nouvel ensemble d'individus (descendance). Celle-ci contient le même nombre d'individus que la population courante. Une fois la descendance obtenue, celle-ci est fusionnée avec la population courante (ses parents). À ce stade, la population unifiée a une taille double de la population initiale. L'opérateur de sélection des parents permet de sélectionner seulement la moitié des individus afin de constituer les parents de la prochaine génération.

L'utilisation des opérateurs génétiques nécessite certaines opérations préalables comme des tris des individus et l'évaluation des fonctions objectif pour chaque individu. L'algorithme 1 montre en détail l'enchaînement de ces opérations.

Algorithme 1 Sélection des parents - concepts communs à NSGA-II et NSGA-III

```
function main
    pop ← new Population                          ▷ Create the initial population
    pop ← randomInitialization(pop)
    while not endCriteria do
        children ← crossoverAndMutation(pop)      ▷ Generate offspring population
        unifiedPop ← pop ∪ children
        evalObjectives(unifiedPop)                ▷ Eval all objectives for each individual
        fronts ← fastNonDominationSort(unifiedPop)
        parents ← selectParents(fronts)
        pop ← parents                             ▷ Set next generation parents
    end while
    return pop
end function
```

Après l'initialisation aléatoire de la population initiale, on entre dans une itération conditionnée par un ou plusieurs critères d'arrêt. Ces critères peuvent être d'ordre temporel (un temps limite d'exécution), ou bien liés au nombre de générations effectuées ou bien au degré de satisfaction des objectifs. Lorsque le ou les critères d'arrêt sont validés, on sort de l'itération et on retourne la population courante. Cette population contient alors les meilleurs individus pour le temps écoulé. Cette population n'est pas ordonnée, pour obtenir les meilleurs individus il faut donc la parcourir. À l'intérieur de l'itération on trouve toutes les étapes nécessaires à la réalisation d'une génération. Tout d'abord on utilise les opérateurs génétiques de croisement et de mutation pour engendrer une descendance depuis la population courante. L'utilisation de ces opérateurs (croisement et mutation) peut être précédée d'une étape de sélection comme dans le cas de la variante NSGA-II (cf. Section 6.4.2). La population courante *pop* et la descendance *children* sont ensuite unifiées en une seule population $unifiedPop$. À ce stade on dispose d'une population du double de la taille de la population courante *pop*. Toutes les fonctions objectif sont évaluées sur chaque individu de cette population unifiée $unifiedPop$. Une fois que les valeurs de chaque fonction objectif sont associées à chaque individu, on utilise la fonction de tri par non domination *fastNonDominationSort()*. Cette fonction permet de ranger les individus par fronts de non domination. Ce tri se déroule comme suit : tous les individus non dominés de la population se voient affectés d'une valeur de rang égal à 0. Ces individus sont enlevés de la population et les individus qui ne sont maintenant plus dominés par aucun des autres se voient affectés de la valeur 1. On itère jusqu'à ce que l'on obtienne un ensemble vide. Les individus qui sont de rang 0 font partie du premier front $\mathcal{F}_0$, les individus de rang 1 font partie du second front $\mathcal{F}_1$ et ainsi de suite. Le tri en fronts de non domination est ensuite utilisé pour sélectionner les parents *parents*. Les parents sont les individus qui constitueront la prochaine population courante *pop*. Cette méthode de sélection des parents diverge beaucoup d'une variante à l'autre. La population courante *pop* se voit donc affectée avec les parents sélectionnés (*parents*) puis on continue l'itération.

La mutation est un opérateur qui peut s'appliquer soit en parallèle avec le croisement (pas de croisement avant mutation) soit après le croisement. Dans tous les cas, la mutation affecte aléatoirement une partie des individus selon un pourcentage prédéfini (paramètre de l'algorithme). Alors que le croisement permet d'obtenir de nouveaux individus parmi leur voisinage, la mutation permet, quant à elle, d'explorer de nouvelles parties de l'espace de recherche (avec un facteur aléatoire). Une fois qu'une descendance est créée, celle-ci est fusionnée avec la population courante (dont les enfants descendent).

6.4.2 La variante NSGA-II

L'algorithme Non-dominated Sorting Genetic Algorithm dans sa deuxième variante (NSGA-II), est un algorithme génétique évolutionnaire de référence [DAPM02b]. Il a su au fil du temps montrer son efficacité et est maintenant utilisé depuis de nombreuses années dans divers domaines et notamment celui du génie logiciel[HMZ12].

Dans NSGA-II, avant d'appliquer les opérateurs génétiques de croisement et de mutation, une première sélection sur la population initiale est réalisée. Cette première sélection permet de choisir les individus sur lesquels on réalisera des croisements. La sélection est réalisée par tournois binaires. Deux individus sont aléatoirement pris dans la population et

le meilleur des deux est conservé. On aura donc autant de tournois binaires que nécessaire pour obtenir la bonne taille de population. La notion de meilleur individu se base sur deux métriques distinctes. Tout d'abord avec le rang (front) qui détermine la qualité de l'individu puis avec la *distance de foule* (crowding distance) qui indique la densité d'individus autour de lui (Algorithme 8). Le rang est l'indicateur prioritaire lors du choix. Si les deux individus ont le même rang alors on conservera celui qui a une densité d'individus alentours la plus faible, autrement dit celui qui est le plus isolé (cf. Section 6.4.5).

6.4.3 La variante NSGA-III

NSGA-II souffre de certaines faiblesses et il est souvent distancé par d'autres algorithmes évolutionnaires [Lie09]. Cependant, depuis ce constat récent (aux alentours de 2010), un article présentant une nouvelle variante de NSGA a été publié par un des auteurs de NSGA-II. Cette nouvelle variante tente de combler certaines faiblesses de la version précédente. NSGA-III [DJ14, JD14b]) est un algorithme évolutionnaire basé sur son prédécesseur NSGA-II. Le principe général de fonctionnement est le même pour les deux algorithmes (cf. Section 6.4.1 et Figure 6.1). Cependant il diverge sur deux points par rapport à la version précédente.

La première sélection effectuée dans NSGA-II avant d'appliquer le croisement et la mutation n'existe plus. Cette sélection permettait de faire une première sélection des « meilleurs » individus par le biais de tournois binaires. Dans NSGA-III, tous les individus ont autant de chances d'être impliqués dans un croisement ou une mutation. Le second et principal point de divergence se trouve au niveau de la sélection des parents. Cette étape est primordiale puisque c'est elle qui permet de choisir les individus qui vont continuer à exister. De la même manière que pour NSGA-II, l'application des opérateurs génétiques engendre une nouvelle population appelée descendance. Cette population est de la même taille que la population des parents. Nous avons vu précédemment que la sélection des parents se fait sur l'union des enfants (descendance) et des parents. Étant donné que cette sélection permet de constituer la nouvelle population de parents, seule la moitié des individus doit être sélectionnée. La sélection est toujours réalisée grâce aux fronts de non domination insérés dans leur intégralité jusqu'à ce qu'un front soit trop grand pour rentrer complètement, soit jusqu'à ce que la population soit pleine. S'il reste des places à remplir avec des individus d'un front de taille trop grande pour rentrer intégralement, alors un choix doit être fait et NSGA-III se démarque par rapport à son prédécesseur sur ce choix.

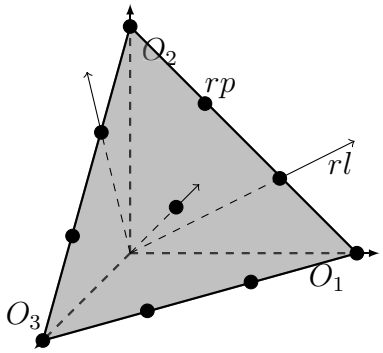


FIGURE 6.2 – NSGA-III hyperplan, points et lignes de références

La sélection des parents dans NSGA-III utilise ce que l'on appelle des points de références. Il s'agit d'un ensemble de points insérés dans l'espace n-dimensionnel des fonctions objectif normalisées, avec n le nombre de fonctions objectif. Les points de référence permettent de localiser les individus dans l'espace. Le nombre de points de référence dépend d'un nombre de divisions p que l'on choisit (paramètre). Ces points sont uniformément répartis sur l'hyperplan (de l'espace n-dimensionnel) qui est construit en reliant tous les points extrêmes ([DJ14, JD14b], Section IV partie C) des axes. Par exemple, pour trois dimensions, si les valeurs maximales pour les trois fonctions objectif valent 1, on aura le triangle (0,0,1)(0,1,0)(1,0,0). Le nombre de divisions correspond au nombre de segmentations de chaque axe (cf. Figure 6.2). La ligne de référence rl est une demi-droite partant de l'origine du repère et passant par un point de référence. Chaque point de référence donne ainsi lieu à une ligne de référence. Ces demi-droites sont un outil permettant d'associer les individus à leur point de référence le plus proche en utilisant la projection orthogonale sur la ligne de référence. Les points de référence permettent ainsi d'avoir une vue sur la répartition des individus dans l'espace des fonctions objectif.

Avant d'associer les individus à leur point de référence, il faut les insérer dans l'espace après leur avoir fait subir une translation vers l'origine du repère. Chaque composante du vecteur de translation correspond à une fonction objectif. Elle est définie par la valeur minimale atteinte par un individu de la fonction objectif associée. Cette valeur minimale est plus précisément déterminée par un meilleur individu du premier front. Le point construit à partir des valeurs minimales sur chaque axe est appelé point idéal.

Pour plus de détails, notamment sur la normalisation des fonctions objectif et la construction de l'hyperplan, nous invitons le lecteur à lire la section IV de la publication originale réalisant une proposition d'algorithme [DJ14, JD14b].

L'orientation prise par NSGA-III par rapport à son prédécesseur tend notamment à augmenter la diversité des individus au sein d'une population afin de favoriser des croisements génétiques plus riches. Cette augmentation de la diversité des individus est obtenue au prix d'une complexité bien plus grande de l'opérateur de sélection des parents.

Les sections suivantes présentent d'un point de vue algorithmique la mise en œuvre qui a été réalisée sur différents points clés de ces algorithmes notamment les opérateurs génétiques mais également la méthode de tri des individus d'une population.

6.4.4 Le tri par fronts de non domination

L'évolution d'une population ne se limite pas à des croisements entre individus et à quelques mutations intervenant aléatoirement sur un petit nombre d'individus. Une sélection doit être périodiquement réalisée afin de limiter l'expansion de la population. On peut se représenter cette sélection un peu à la manière d'une sélection naturelle où les

individus les plus faibles seraient amenés à disparaître. Par ce parallèle on déduit aisément que nous allons avoir besoin de classer (trier) les individus de notre population en fonction de leur « force/faiblesse ».

NSGA utilise un tri qui se base sur la domination des individus par rapport aux autres. Un individu qui n'est pas dominé sera par conséquent jugé « meilleur » qu'un individu qui l'est. La relation de domination est donc primordiale pour classer les individus et cette dernière est donc établie pour l'ensemble des individus de la population à trier.

Algorithme 2 Tri rapide en fronts de non domination

```
function fastNonDominatedSort(pop)
    fronts ← ∅
    for i in range [0, pop.size[ do
        pop[i].isDomCount ← 0
        pop[i].domSet ← ∅
        for j in range [0, pop.size[ do
            if dominates(pop[i], pop[j]) then
                pop[i].domSet.add(pop[j])
            else if dominates(pop[j], pop[i]) then
                pop[i].isDomCount ← pop[i].isDomCount + 1
            end if
        end for
        if pop[i].isDomCount = 0 then
            pop[i].rank ← 1
            fronts[0].add(pop[i])
        end if
    end for
    current ← 0
    repeat
        for i in range [0, fronts[current].size do
            for j in range [0, fronts[current][i].domSet.size[ do
                p2 ← fronts[current][i].domSet[j]
                p2.isDomCount ← p2.isDomCount − 1
                if p2.isDomCount = 0 then
                    p2.rank ← current + 1
                    front.add(p2)
                end if
            end for
            fronts[current][i].domSet ← ∅
        end for
        if front.size then
            fronts.add(front)
        end if
        current ← current + 1
    until current < fronts.size
    return fronts
end function
```

Le tri de la population se fait sous la forme de fronts appelés fronts de non domination. Pour réaliser ce tri nous avons tout d'abord besoin d'affecter un rang à nos individus. Les individus qui ne sont dominés par aucun autre ont le rang 0. Si on fait maintenant abstraction des individus affectés du rang 0. L'ensemble des individus (restants) qui ne sont dominés par aucun autre se voient affectés du rang 1. En faisant maintenant abstraction du rang 0 et du rang 1, les individus qui ne sont dominés par aucun autre sont affectés du rang 2 et ainsi de suite jusqu'à que l'ensemble des individus soit classé. La création des fronts de non domination devient immédiate. Le premier front contient les individus affectés du rang 0, le second front contient les individus affectés du rang 1 etc. L'algorithme 2 donne un aperçu de la construction des fronts de non domination pour une population donnée.

La domination d'un individu par rapport à un autre, ou plutôt la non domination d'un individu par rapport à un autre, puisque dans notre cas nous cherchons à trouver les individus non dominés, se définie selon la formule suivante :

$$i \text{ domine } j \;\text{ ssi }\; \forall\, f \in fcts.\, Obj,\; i, j \in population,\; f(i) > f(j)$$

Tel que défini ici, un individu domine un autre si et seulement s'il est meilleur sur l'ensemble des fonctions objectif. Cette formule nous permet donc de savoir si un individu en domine un autre. Elle sera donc utilisée deux fois par paire d'individus afin de savoir si le premier individu domine le second mais également, dans le cas ou ça ne sera pas le cas, si le second domine le premier. En effet, si le premier individu ne domine pas le second, cela ne signifie pas forcément que le second domine le premier. Deux individus qui sont mutuellement meilleurs que l'autre sur des fonction objectifs différentes seront alors incomparables.

6.4.5 Les opérateurs génétiques

La conception des opérateurs génétiques peut varier. Des versions différentes existent notamment pour le croisement selon que l'on considère un encodage réel ou binaire des individus. Un problème à variables réelles est un problème où les domaines des variables contiennent les valeurs non binaires du problème (une chaîne de caractères nommant une couleur, une taille, etc.). À l'inverse, un problème aura une représentation binaire si on utilise une représentation « binaire » des variables. Dans ce dernier cas, les opérateurs de croisement et de mutation opèreront sur chaque bit de la chaîne binaire. Chaque bit peut donc être associé à un gène. Même dans le cas d'un problème à variables réelles, il est possible de simuler un opérateur de croisement binaire. Le choix de conserver un opérateur réel pour traiter des variables a été fait. Nous avons également choisi de garder les opérateurs les plus généraux possible et d'éviter à tout prix de les spécialiser ou de les guider. Plusieurs tests ont montré que le guidage des opérateurs génétiques n'était pas forcément un plus et que cela pouvait amener à des maxima locaux.

Croisements générant un seul enfant

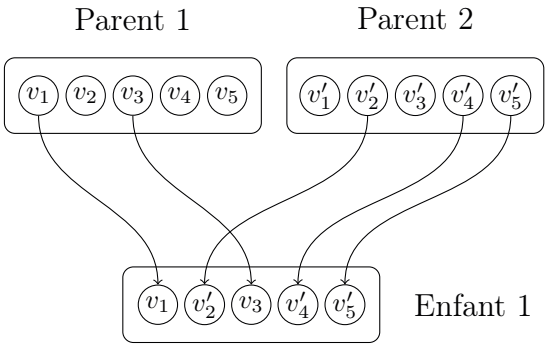


FIGURE 6.3 – Opérateur de croisement donnant un seul enfant

Pour le croisement des individus, on trouve généralement deux grands types d'implémentations : celles qui génèrent un seul enfant à partir de deux parents (cf. Figure 6.3) et celles qui génèrent deux enfants (cf. Figure 6.4) à partir de deux parents. L'Algorithme 3 décrit le croisement de deux individus ne générant qu'un seul enfant. Des parties de chaque parent sont aléatoirement sélectionnées pour constituer l'enfant. Pour chaque valeur de variable, soit on prend la valeur depuis le premier parent, soit on prend cette valeur depuis le second parent. Le choix de prendre la valeur de l'un ou l'autre des deux parents est complètement aléatoire ce qui peut entraîner des cas particuliers. L'enfant ainsi généré peut potentiellement être la copie conforme d'un des deux parents. Mais la probabilité que cela arrive, c'est-à-dire la probabilité de prendre à chaque fois la valeur depuis le même parent est de $\left(\frac{1}{2}\right)^n$ avec n le nombre de variables qui constituent un individu.

Algorithme 3 Croisement donnant un enfant

```
function crossover_single-child(parentA, parentB)
    child ← new individual                          ▷ Create a new individual
    components ← parentA.nbComponents               ▷ Number of genes (variables
    for c in range [0, components[ do               ▷ For each parent component
5:      rndVal ← random()                           ▷ Generate a random value
        if odd(rndVal) then                         ▷ rndVal is odd
            child[c] ← parentA[c]                   ▷ c Component of parent A
        else
            child[c] ← parentB[c];                  ▷ c Component of parent B
10:     end if
    end for
    return child
end function
```

La recopie intégrale d'un individu (clonage) n'est pas problématique dans l'exécution de l'algorithme. D'autres opérateurs comme la sélection peuvent également engendrer des individus identiques dans une même population. De cette manière, lors des croisements, plusieurs individus identiques seront croisés avec des individus différents. Par conséquent un individu en particulier n'est pas condamné à n'être croisé qu'avec un seul autre (puisque des clones de celui-ci peuvent exister).

Croisements générant deux enfants

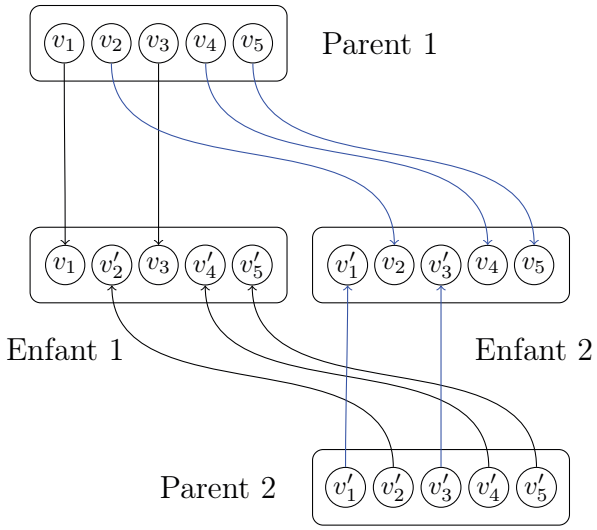


FIGURE 6.4 – Opérateur de croisement donnant deux enfants

Le croisement donnant deux enfants est une autre manière d'implémenter cet opérateur (Algorithme 4). Dans ce cas, à partir des deux parents, deux individus opposés seront constitués. À la différence de l'implémentation précédente, si pour le premier enfant on choisit la valeur v_i depuis le premier parent, on prendra cette valeur depuis le second parent pour constituer l'autre enfant. Autrement dit, pour chaque valeur d'un parent si celle-ci n'est pas utilisée pour constituer le premier enfant alors elle est obligatoirement utilisée pour constituer le second enfant. De la même façon que pour le croisement ne donnant lieu qu'à un seul enfant, il existe aussi un cas particulier où les parents pourraient être clonés. En effet la probabilité est certes très faible mais non nulle pour que l'on choisisse toutes les valeurs depuis un même parent pour constituer un enfant. Si cela se produit, un des deux enfants sera la copie parfaite d'un des deux parents et l'autre enfant sera la copie de l'autre parent. Ici aussi la composante aléatoire représente une part importante de l'algorithme.

Algorithme 4 Croisement générant deux enfants

```
    procedure crossover_two-children(parentA, parentB, child1, child2)
        child1 ← new individual                          ▷ Create a new individual
        child2 ← new individual                          ▷ Create a new individual
        components ← parentA.nbCompoents                 ▷ Number of genes (variables
 5:     for c in range [0, components[ do                ▷ For each parent component
            rndVal ← random()                            ▷ Generate a random value
            if odd(rndVal) then                          ▷ rndValue is odd
                child1[c] ← parentA[c]                   ▷ c Component of parent A
                child2[c] ← parentB[c]                   ▷ c Component of parent B
10:         else
                child1[c] ← parentB[c]                   ▷ c Component of parent B
                child2[c] ← parentA[c]                   ▷ c Component of parent A
            end if
        end for
15: end procedure
```

Mutation

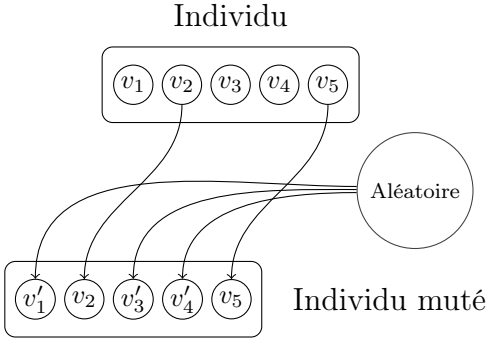


FIGURE 6.5 – Opérateur de mutation à base aléatoire

Alors que l'opérateur de croisement (qu'il génère un seul ou deux individus) permet de mélanger les individus entre eux pour en engendrer de nouveaux, l'opérateur de mutation (Algorithme 5) permet, quant à lui, de faire apparaître de nouvelles valeurs. La mutation altère une partie ou l'intégralité d'un individu. Cette transformation s'inspire de la mutation d'un ou plusieurs gènes dans un génome. Ce sont les mutations génétiques qui permettent l'évolution (bénéfique ou non) d'une espèce en permettant l'apparition de nouveaux gènes. Ces nouveaux gènes viendront se mélanger au milieu des autres gènes et interagiront avec eux grâce aux opérateurs génétiques. Encore une fois la part d'aléatoire est la base de cet opérateur. Chacune des variables aura une certaine probabilité (définie) de muter. Et lorsqu'une variable mute, la valeur est également choisie aléatoirement dans le domaine de définition correspondant. Dans notre mise en œuvre, les valeurs sont choisies sur une base d'équiprobabilité sur l'ensemble du domaine de définition.

Algorithme 5 Opérateur de mutation d'un individu

```
    function mutate(individual)
        individual2 ← new individual                    ▷ Create a new individual
        components ← parentA.nbCompoents                ▷ Number of genes (variables
        for c in range [0, components[ do               ▷ For each parent component
 5:         rndVal ← random()                           ▷ Generate a random value
            if odd(rndVal) then                         ▷ rndVal is odd
                individual2[c] ← rndValForComponent(c)  ▷ Rnd val in c component domain
            else
                individual2[c] ← individual[c]          ▷ Copy original value
10:         end if
        end for
        return indivuall2
    end function
```

Cette implémentation très simple de l'opérateur de mutation peut, de la même manière que pour le croisement, amener à des cas particuliers. En effet, si $rndVal$ est toujours impair, alors l'individu n'aura subi aucune mutation. De plus, dans le cas où $rndValue$ serait pair, alors il est très peu probable mais possible que l'individu ne soit toujours pas muté. Le fait de prendre une valeur aléatoire dans le domaine de définition de la composante peut très bien donner la même valeur que celle qui est actuellement choisie. Ces cas de figures ne sont pas problématiques dans le sens où il n'ont que très peu de probabilités d'arriver. La probabilité qu'un individu ne mute pas est d'au maximum $\left(\frac{1}{2}\right)^n$ avec n le nombre de composantes d'un individu. Ce qui donne une probabilité d'au maximum 6.25% pour 4 composantes et d'au maximum 0.1% pour 10 composantes. Dans le cas de pages réelles, comme c'est le cas dans cette expérimentation, le nombre de composantes (nos variables) est de quelques dizaines voire de quelques centaines dans le cas de pages complexes.

Sélection

L'opérateur de sélection simple n'est utilisé que dans NSGA-II. Il permet de réaliser, depuis la population initiale, une sélection préalable des individus qui vont être croisés et potentiellement mutés. Cette sélection est réalisée par « tournois binaires » (Algorithme 6). Pour chaque paire d'individus on garde le meilleur en terme de rang et de « crowding distance » (distance de foule).

Algorithme 6 Sélection par tournois binaires

```
function BinarySelection(population, popSize)
    selectedPopulation ← ∅
    for i in range [0, popSize[ do
        rndIndA ← rndInd(population)
        rndIndB ← rndInd(population)
        selectedPopulation.add(better(rndIndA, rndIndB))
    end for
    return selectedPopulation
end function
```

Algorithme 7 Retourne le meilleur des deux individus

```
function better(individual1, individual2)
    if individual1.rank = individual2.rank then          ▷ If true, get the most isolated
        if individual1.crowdingDistance > individual2.crowdingDistance then
            return individual1
        else
            return individual2
        end if
    else                                                 ▷ Get which one with the lowest rank
        if individual1.rank < individual2.rank then
            return individual1
        else
            return individual2
        end if
    end if
end function
```

L'utilisation de la sélection nécessite que le rang et la distance de foule soient préalablement calculés pour chaque individu. Ce calcul se fait notamment au moment du tri par front de non domination. Le rang correspond à l'indice du front où se trouve l'individu. La distance de foule représente quant à elle, le niveau d'isolement d'un individu à l'intérieur du front dans lequel il se trouve. Plus la distance est importante et plus ce dernier sera isolé des autres individus du front. La distance est calculée dans un espace n dimensionnel où n est le nombre de fonctions objectif. Par conséquent, on utilise les valeurs retournées par les fonctions objectif afin de placer les points de l'espace n dimensionnel pour ensuite calculer leurs distances.

L'opérateur de sélection implémenté conserve, lorsque le rang des deux individus est identique, l'individu le plus isolé (dans son front). Ceci permet de conserver une certaine diversité dans la population.

Algorithme 8 Calcul de la distance de foule

```
    procedure CrowdingDistance(pop, nbObjs)
        popSize ← pop.size                                  ▷ For simplification purpose
        for ind in range [0, popSize[ do
            pop[ind].crowdDist ← 0.0          ▷ Initialize crowding distance of individual ind
 5:     end for
        for o in range [0, nbObjs[ do
            sortByObjective(pop, o)                  ▷ Sort individuals according objective o
            indMin ← pop[0]                        ▷ The individual with the minimal obj value
            indMax ← pop[popSize − 1]              ▷ The individual with the maximal obj value
10:         range ← indMax.objRate[o] − indMin.objRate[o]
            pop[0].crowdDist ← MAX_FLOAT           ▷ Assign "infinite" distance to min
            pop[popSize − 1].crowdDist ← MAX_FLOAT                    ▷ idem for max
            if range ≠ 0 then
                for k in range [1, popSize − 1[ do
15:                 nVal ← (pop[k + 1].objRate[o] − pop[k − 1].objRate[o]) / range
                    pop[k].crowdDist ← pop[k].crowdDist + nVal
                end for
            end if
        end for
20: end procedure
```

Sélection des parents (NSGA-II)

Nous revenons une nouvelle fois sur la sélection des parents qui est réalisée par le biais des fronts F_i de non domination. Chaque front contient un groupe d'individus « au même niveau », c'est-à-dire qu'aucun ne domine ou n'est dominé par un autre. Le premier front F_0 contient donc tous les individus qui ne sont dominés par aucun autre, le front F_1 contient tous les individus qui ne sont dominés que par ceux de F_0 et ainsi de suite. Afin de construire la population des parents on ajoute un à un les fronts $F_0, F_1, F_2, ...$ Les fronts doivent être ajoutés dans leur intégralité. De ce fait, l'ajout des fronts se fait jusqu'à ce que la population soit entièrement remplie ou bien jusqu'à ce qu'un front ne puisse pas rentrer en entier (pas assez de place dans la population en construction pour y loger tous ses individus). Lorsque des places restent à pourvoir, un choix des individus dans le dernier front F_l doit être réalisé.

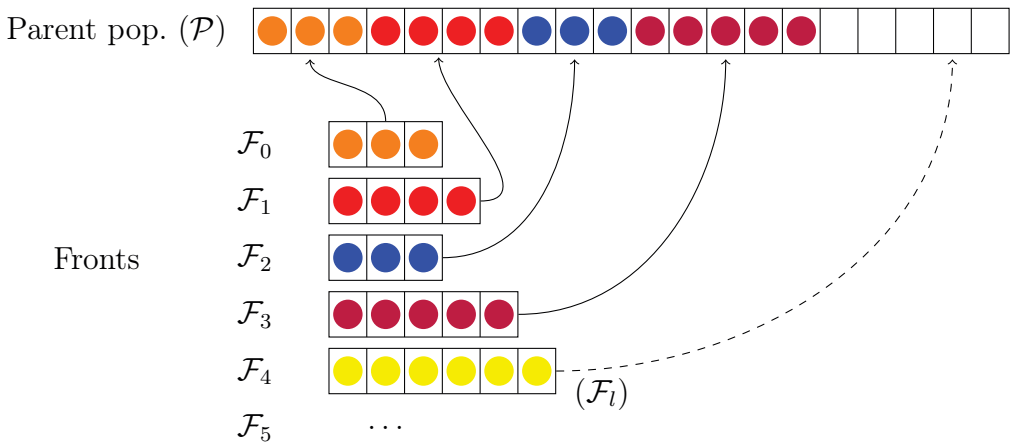


FIGURE 6.6 – NSGA-II et NSGA-III sélection des parents

Dans l'exemple de la figure 6.6, une population d'individus a été triée en fronts de non domination. Cette population provient de l'union des parents de la génération précédente avec les enfants générés. Ces fronts de non domination $\mathcal{F}_i$ sont ensuite utilisés pour remplir la population. Ici les premiers fronts $(\mathcal{F}_0, \mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3)$ rentrent intégralement dans la population $\mathcal{P}$. Lorsque vient le tour de $\mathcal{F}_4$, ce front contient trop d'individus pour tous les loger dans les emplacements libres de $\mathcal{P}$. Il faut donc choisir une partie des individus. Le dernier front que l'on tente d'insérer, et pour lequel il faut faire un choix des individus à prendre, est appelé dernier front et est noté $\mathcal{F}_l$ ('l' pour "last").

Dans NSGA-II, ce choix est réalisé en fonction de la distance de foule. La distance de foule est calculée sur le dernier front $\mathcal{F}_l$. Chaque individu de ce dernier front est affecté de sa distance de foule par rapport à tous les autres individus de ce front. Une fois les distances de foule calculées on sélectionne les individus les plus isolés en premier. Et ce, autant de fois que nécessaire pour remplir la population des parents $\mathcal{P}$.

Sélection des parents (NSGA-III)

La sélection des parents dans NSGA-III, aussi appelée « sélection environnementale », diffère beaucoup de celle de NSGA-II (Voir Algorithme 9). C'est la principale différence entre les deux versions de l'algorithme. Au lieu de se baser sur la distance de foule, on se base sur ce que l'on appelle des points de référence. Ces points sont des repères dans l'espace n-dimensionnel formé par les fonctions objectif normalisées, avec n le nombre de fonctions objectif. Les points de référence sont répartis dans une portion spécifique de l'espace et permettent de localiser des individus avec plus ou moins de précision. Pour un nombre de fonctions objectif donné, plus le nombre de points de référence sera important et plus la précision sera élevée. Les individus sont donc sélectionnés à la fois par leur rang mais aussi selon leur position dans l'espace des fonctions objectif comme présenté précédemment dans la section 6.4.3.

Un grand nombre d'étapes constituent cette sélection qui est au cœur de l'amélioration apportée par NSGA-III. Néanmoins toutes ces étapes n'ont pas de complexité élevée. Un certain nombre d'entre elles font appel à du calcul géométrique dans un espace à plusieurs

dimensions. L'implémentation de ces fonctions ne diverge pas de celle de l'article originel et de l'implémentation non officielle proposée [1] qui complète l'article.

Algorithme 9 Sélection environnementale

```
function environmentalSelection(pop, popSize, nbObjs)
    parents ← new population() ;
    fronts ← fastNonDominatedSort(pop) ;
    last ← 0 ;
    nextSize ← fronts[0].size
    while nextSize ≤ popSize do
        for i in range [0, fronts[last].size[ do
            parents.add(fronts[last][i])
        end for
        if nextSize = popSize then
            return parents
        end if
        last ← last + 1
        nextSize ← nextSize + fronts[last].size
    end while
    inserted ← nextSize − fronts[last].size
    delete(fronts, last + 1, END)          ▷ Delete useless fronts from last+1 to the end
    idealPoint ← translateObjectives(nbObjs, fronts)          ▷ Translate objs scores
    findExtremePoints(extremePoints, nbObj, fronts)
    intercepts ← ∅
    constructHyperPlane(intercepts, nbObj, extremePoints)
    normalizeObjectives(nbObj, fronts, intercepts, idealPoint)
    associate(rpts, fronts)
    while inserted < popSize do
        minRpt ← findNicheReferencePoint(rpts)          ▷ min assoc. individuals
        chosen ← selectClusterMember(minRpt)          ▷ Assoc. individual with min dist.
        if chosen = ∅ then          ▷ No potential member in Fl
            rpts.delete(minRpt)          ▷ Disregard this reference point
        else
            minRpt.increaseMembersCount()
            minRpt.rmPotentialMember(chosen)
            parents.add(chosen)
            inserted ← inserted + 1
        end if
    end while
    for d in range [0, fronts.size[ do
        delete(fronts[d])          ▷ Clear remaining fronts from list
    end for
    return parents ;
end function
```

1. http://web.ntnu.edu.tw/~tcchiang/publications/nsga3cpp/nsga3cpp.htm

6.4.6 Les fonctions objectif

La représentation en machine des préférences est réalisée par les fonctions objectif. Ces fonctions permettent d'évaluer des préférences définies sur un ou plusieurs éléments de la page HTML. On peut définir des fonctions simples comme une comparaison de taille de police, mais les fonctions objectif permettent d'implémenter des préférences bien plus complexes. Nous détaillerons ici les algorithmes de certaines fonctions objectif non triviales. Trois types de fonctions objectif implémentant des préférences basées sur les couleurs et leurs propriétés physiques vont être décrites ainsi que le type de valeurs retournées par ces fonctions.

Valeurs retournées

On trouve souvent des implémentations d'algorithmes génétiques qui utilisent des fonctions retournant une valeur binaire. Mais ce n'est pas la seule manière d'aborder le problème. NSGA-II et NSGA-III utilisent tous deux de nombreux tris en fonction des valeurs de fonctions objectif retournées. Ces valeurs sont utilisées pour le tri en front de non domination. Si on utilise des valeurs binaires comme par exemple 0 non satisfait et 1 satisfait, alors beaucoup d'individus ne sont pas comparables. Nous utilisons donc des fonctions objectif retournant des valeurs à virgule flottante. Celles-ci permettent notamment d'avoir une plus grande précision sur le niveau de satisfaction. De cette manière il est possible d'indiquer qu'un individu ne satisfait pas la préférence mais que ce n'est pas le pire. On est donc en mesure, en comparant les valeurs retournées, de distinguer entre deux individus satisfaisants lequel satisfait davantage la préférence. Les calculs de la satisfaction retournent des valeurs dans les intervalles suivants ($prefNb$ est le nombre de préférences) :

$$[0,\ 0+\epsilon[\ et\ [1,\ 1+\epsilon[\qquad Avec \qquad \epsilon = \frac{1}{prefNb} \tag{6.12}$$

Dans cette formule, ϵ est une borne très petite qui limite la valeur s'ajoutant à 0 lorsque la préférence n'est pas satisfaite ou à 1 dans le cas contraire. Cette borne ϵ est définie en fonction du nombre de préférences que l'on a.

Plus concrètement, supposons qu'un utilisateur définisse la préférence « taille de police au moins égale à 12pt ». Si on a une taille de police qui est de 12pt alors on satisfait la préférence mais on la satisfait tout juste. Dans ce cas-là on retournera 1.0. Maintenant si on a une taille de police de 14pt, on peut estimer que par la définition de cette préférence, 14pt est meilleure que 12pt, alors on retourne par exemple 1.001 et 1.0011 pourrait être retourné pour 15pt. Autre exemple, si l'utilisateur veut que ses liens soient rouges, alors si on a un lien qui est presque rouge, même s'il ne satisfait pas la préférence, c'est toujours mieux qu'un lien bleu. Cette méthode permet donc d'exprimer cela.

La limitation des valeurs retournées par ϵ a son importance lorsque l'on combine les préférences utilisateur. L'agrégation permet de combiner plusieurs préférences utilisateur en une seule. Dans ce cas-là une seule valeur de retour doit être retournée pour toutes les préférences. Si l'on fait la somme des valeurs de retour sans précaution alors on pourrait obtenir une erreur dans le nombre de fonctions satisfaites. Avec des valeurs binaires cela

ne pose pas de souci, par exemple si on a trois préférences et que les valeurs de retour sont 0, 0 et 1 alors la somme des trois donne 1, correspondant au fait qu'il y a bien une seule préférence satisfaite. Mais avec des valeurs à virgule flottante c'est un peu plus délicat. Si on a cette fois les trois valeurs 0.5, 1 et 0.5 alors la somme des trois valeurs de préférence donne 2 et pourtant une seule est satisfaite. C'est notamment pour cette raison que la décimale ajoutée pour augmenter la précision est bornée.

Uniformisation des brillances d'arrière-plan

Une préférence d'uniformisation des brillances d'arrière-plan permet principalement de limiter les effets d'éblouissement dus à des arrière-plans lumineux proches d'arrière-plans sombres. Autrement dit, il s'agit d'éviter d'avoir à la fois des arrière-plans sombres et des arrière-plans lumineux sur la même page.

Algorithme 10 Fonction objectif - Uniformité des brillances de couleur

```
1:  function eval_uniformBrightness({v_1, v_2, .., v_n}, maxDiff)
2:      min ← min(getLum(v_1), getLum(v_2),.., getLum(v_n))   ▷ Get min brightness
3:      max ← max(getLum(v_1), getLum(v_2),.., getLum(v_n))   ▷ Get max brightness
4:      d ← max − min                                  ▷ Difference, between 0 and 1
5:      if d ≥ maxDiff then                                    ▷ If not satisfied
6:          return 0 + (1 − d)/prefNb
7:      else                                                       ▷ If satisfied
8:          return 1 + (1 − d)/prefNb
9:      end if
10: end function
```

Il existe plusieurs façons d'évaluer cet objectif. Il s'agit ici d'une préférence globale sur tous les arrière-plans, mais elle est néanmoins décomposable. Ainsi, il est possible de décomposer cette dernière en fonctions objectif binaires distinctes liant tous les couples possibles de variables. L'algorithme réalisé traite ce problème de manière globale (cf. Algorithme 10). Avoir une brillance d'arrière-plan uniforme revient à dire que chaque arrière-plan de la page a une brillance proche de tous les autres. Ceci peut être réalisé par la simple comparaison de la brillance la moins élevée avec celle qui est la plus élevée. De cette manière, si cette différence est inférieure au seuil défini par l'utilisateur, nous avons la garantie d'une uniformité des brillances (relativement au seuil défini), puisque l'on compare les deux éléments extrêmes.

Pour utiliser cette fonction on a besoin de calculer la brillance (ou luminance) des couleurs à comparer. Ce calcul est réalisé par la fonction *getLum*. Le calcul de la brillance d'une couleur est défini dans le Web Content Accessility Guidelines (WCAG 2.0 [WCA]). Ce calcul a été extrait et est détaillé en section 2.4.2. Il se base sur la perception lumineuse de l'œil humain qui diffère selon la composante de couleur.

Telle qu'elle est implémentée, la fonction permet simplement d'évaluer l'uniformité de brillance d'une série de couleurs. Cependant même si jusque là nous avons utilisé cette fonction pour mesurer l'uniformité de brillance des arrière-plans, cette fonction n'est pas liée à un type d'éléments. De ce fait elle peut très bien être utilisée pour mesure l'uniformité de brillance d'éléments d'avant-plan ou plus généralement de tout éléments colorés.

Contraste de brillance minimal

La préférence de contraste minimal de brillance permet notamment d'augmenter la lisibilité et le confort de lecture d'un texte vis-à-vis de son arrière-plan direct. De la même manière que pour la fonction précédente l'évaluation de cette fonction nécessite l'utilisation de la fonction *getLum* permettant d'obtenir la brillance (luminance) d'une couleur.

Algorithme 11 Fonction objectif - Contraste de brillance minimum

```
function eval_minContrast(color1, color2, minContrast)
    l1 ← getLum(color1)
    l2 ← getLum(color2)
    contrast ← floor((max(l1, l2)+0.05) / (min(l1, l2)+0.05) × 10)10   ▷ From 0 to 21
    contrast = contrast ∗ 100/21                                       ▷ Map to 0-100%
    diff = (contrast − minContrast)/100                                ▷ Get diff between 0 and 1
    if contrast ≥ minContrast then                                     ▷ If satisfied
        return 1 + (diff)/prefNb
    else                                                               ▷ If not satisfied
        return 0 + (diff)/prefNb
    end if
end function
```

Cette fonction objectif compare le niveau de contraste calculé entre les deux éléments concernés et le seuil minimal renseigné *minContrast*. Le contraste de brillance correspond au rapport de la brillance la plus élevée sur la brillance la plus faible, ramené entre 0 et 100%. Ce rapport est donné par le WCAG 2.0.

Distance de couleurs

La distance entre deux couleurs est une valeur permettant de qualifier à quel point les couleurs sont différentes. Dans notre cas, nous utilisons la teinte de la couleur comme élément de comparaison. La fonction objectif permettant d'implémenter cette préférence prendra par conséquent deux paramètres : la couleur 1 et la couleur 2. Cette fonction tient compte uniquement de la teinte de la couleur. L'utilisation de la saturation permet d'avoir une indication sur la différence entre les couleurs d'une même teinte ce qui rajoute plus de détails. De plus dans l'implémentation qui a été réalisée, lorsque le niveau de saturation est très bas on considère que toutes les teintes sont identiques, ce qui augmente le nombre de combinaisons satisfaisant la contrainte de proximité de couleurs. Cependant le seuil de saturation en-dessous duquel on peut considérer toutes les teintes comme étant proches n'est pas évident à fixer et dépend de la sensibilité aux couleurs de chacun. Ce seuil doit donc d'une manière ou d'une autre, explicitement ou implicitement, être défini par l'utilisateur. On pourrait encore augmenter la précision en utilisant également la composante de luminance (brillance) de la couleur.

Dans notre cas avons dérivé une fonction « classique » de conversion de sRGB vers HSV (Hue, Saturation, Value) afin de ne conserver que la conversion vers la composante de teinte H. L'algorithme suivant (cf. 12) décrit l'implémentation de la fonction permettant d'extraire la teinte depuis une couleur sRGB.

Algorithme 12 Teinte à partir d'une couleur sRGB

```
function getHue(red, green, blue)
    maxVal ← max(red, green, blue)
    minVal ← min(red, green, blue)
    delta ← maxVal − minVal
    if delta = 0 then
        return 0
    end if
    H ← 0                                                    ▷ The hue variable
    delta_R ← (((maxVal − red)/6.0) + (delta/2.0))/delta
    delta_G ← (((maxVal − green)/6.0) + (delta/2.0))/delta
    delta_B ← (((maxVal − blue)/6.0) + (delta/2.0))/delta
    if red = maxVal then
        H ← delta_B − delta_G
    else if green = maxVal then
        H ← 0.333 + delta_R − delta_B
    else if blue = max then
        H ← 0.666 + delta_G − delta_R
    end if
    if H < 0 then
        H ← H + 1
    else if H > 1 then
        H ← H − 1
    end if
    return H × 360.0
end function
```

L'Algorithme 13 décrit l'implémentation de cette fonction objectif. Nous comparons ici la teinte de *curColor* (couleur courante) avec celle de *origColor* (couleur d'origine). Si l'angle minimum entre les deux teintes est inférieur au seuil prédéfini alors la fonction objectif est satisfaite. Dans le cas contraire elle sera non satisfaite.

Algorithme 13 Fonction objectif - Distance de couleur maximale

```
function eval_maxColorDistance(curColor, origColor, maxDist)
    origHue ← getHue(origColor)                          ▷ Get original color hue
    curHue ← getHue(curColor)                            ▷ Get current color hue
    hueDiff ← curHue − origHue
    if hueDiff ≥ 180 then
        hueDiff ← 360 − hueDiff                          ▷ Set hue diference <= 180
    end if
    hd ← hueDiff/180                                     ▷ Get hueDiff between 0 and 1
    if hueDiff < maxAngle then                           ▷ If satisfied
        return 1 + (hd)/prefNb
    else                                                 ▷ If not satisfied
        return 0 + (hd)/prefNb
    end if
end function
```

L'angle entre les deux couleurs est borné entre 0° et 180°. Lorsque l'angle est égal à 180° les deux couleurs se trouvent donc à l'opposé l'une de l'autre sur le cercle chromatique. L'angle de différence entre les deux couleurs est l'angle minimal entre ces couleurs. Si, par exemple, la différence donne un angle de 270°, alors, uniquement 90° séparent les deux couleurs en tournant dans l'autre sens (cf. cercle chromatique présenté en section 2.10). Par conséquent si l'angle obtenu par la différence est supérieur à 180° on le ramène entre 0° et 180° « en tournant dans l'autre sens ».

Cette fonction peut être dérivée afin de calculer une distance minimale en fixant à la place du seuil maximal un seuil minimal. Cette dérivée permettrait par exemple d'assurer une différence minimale entre deux couleurs selon leurs teintes ce qui diffère de la fonction de contraste de brillance basée sur le rapport des brillances perçues des deux couleurs.

Agrégation des fonctions objectif

L'agrégation des objectifs au travers de leur fonction objectif est un cas un peu à part. Cette technique est utilisée dans les algorithmes évolutionnaires n'ayant qu'une seule fonction objectif. Cependant rien n'exclut son utilisation dans un algorithme multi-objectifs. Un des avantages d'utiliser des fonctions objectif agrégées dans un algorithme multi-objectifs est qu'on n'est pas forcé de tout agréger en une seule. On peut donc disposer de plusieurs fonctions objectif agrégées.

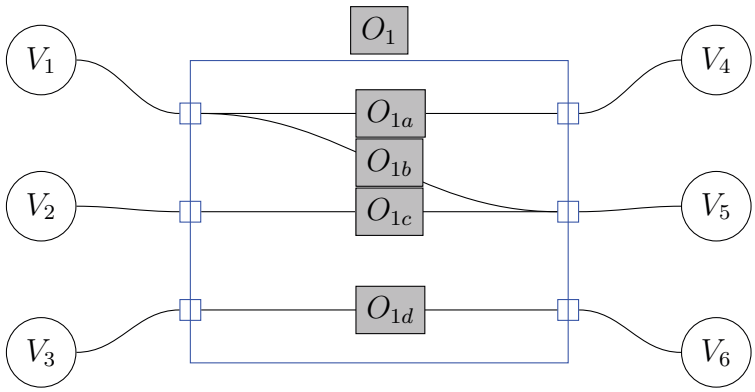


FIGURE 6.7 – Exemple d'agrégation des fonctions objectif

L'opérateur d'agrégation utilisé par défaut utilise la moyenne. Cette opération permet à la fois de savoir si toutes les fonctions objectif agrégées sont satisfaites mais aussi d'obtenir une valeur à virgule flottante donnant la satisfaction de la fonction objectif agrégée. Dans l'implémentation réalisée, les fonctions objectif restent évaluées indépendamment, puis leurs résultats sont combinés. Une trace de toutes les préférences est par conséquent conservée. L'intérêt de l'agrégation des fonctions objectif à la manière d'un algorithme mono-objectif, réside dans le fait de réduire la complexité de l'algorithme. Cette complexité étant de MN^2 avec N la taille de la population et M le nombre de fonctions objectif.

6.5 Conclusion

Le nombre de possibilités de transformation d'une page Web est gigantesque. Il est donc clairement impossible de parcourir toutes les possibilités une par une jusqu'à trouver une adaptation qui satisfait toutes les préférences de l'utilisateur. Nous avons tout d'abord étudié les algorithmes issus de la théorie des préférences puisque cette dernière semble être fortement en lien avec la recherche d'une adaptation selon des préférences de l'utilisateur. Cependant, étant donné notre espace de recherche mais aussi les dépendances et conflits entre nos préférences, il n'est pas possible d'utiliser ces algorithmes. Nous avons donc étudié plus en profondeur une approche assez différente. Les algorithmes génétiques permettent de manipuler des solutions potentielles (individus) dans des espaces de recherche très grands. De bons résultats ont été obtenus avec différentes versions d'algorithmes génétiques. Les algorithmes issus de la théorie des préférences ont fait l'objet d'implémentations afin d'appuyer les hypothèses théoriques concernant les limites de cette approche. Deux versions de l'algorithme génétique NSGA ont également été implémentées. Le chapitre suivant (cf. Chapitre 8) mène une expérimentation sur plusieurs sites Web réels de natures différentes afin de comparer les résultats des diverses approches. D'autres approches comme la programmation par contraintes ou le Deep Learning pourraient potentiellement donner de bons résultats, ces deux approches méritent donc d'être abordées et font partie des futurs travaux de recherche.

Chapitre 7

Schéma global du démonstrateur

Préambule

La recherche d'une adaptation passe par le traitement des éléments HTML de la page afin de satisfaire les préférences de l'utilisateur. Ce chapitre décrit le fonctionnement global du démonstrateur mis en place afin de réaliser l'adaptation de pages Web. Le démonstrateur a été construit autour des deux versions (II et III) de NSGA (Non-dominated Sorting Genetic Algorithm). Nous décrivons donc ici les diverses structures de données utilisées et les paramètres des algorithmes génétiques. Ainsi nous dessinons le processus global d'adaptation de la page Web à l'aide des préférences de l'utilisateur.

Sommaire

7.1 Sélection des données depuis la page Web **115**
7.2 Données en entrée **116**
7.3 Données en sortie et transformation de la page Web **118**
7.4 Architecture générale **118**
7.5 Les différents paramètres **119**
7.6 Exemples d'adaptations **119**
7.7 Conclusion **121**

7.1 Sélection des données depuis la page Web

Les chapitres 4 et 5 donnent tous deux une première intuition des éléments HTML importants sur lesquels vont porter les souhaits de l'utilisateur. C'est en fonction des souhaits de transformation définis que l'on choisit les éléments et leurs propriétés. Par exemple, pour un souhait de taille de police supérieure à une valeur donnée, il faudra extraire tous les éléments textuels de la page ainsi que leur taille actuelle.

À ce stade de développement du démonstrateur, l'extraction, c'est-à-dire le passage de la page Web aux données d'entrée du programme, est réalisée manuellement. La sélection des éléments HTML ciblés par un souhait n'est pas triviale. Des souhaits peuvent très bien cibler des éléments particuliers qui semblent à première vue simples, comme un menu principal d'une page cependant, selon la version de HTML utilisée pour le développement, il sera très difficile de l'identifier dans le code source. Si dans HTML 5 les menus de navigation peuvent être identifiés par une balise qui leur est consacrée, dans les versions antérieures de HTML ce n'est pas le cas. Les menus sont donc fondus au milieu du reste. L'automatisation de cette étape de sélection est possible si une première phase de reconnaissance des structures logiques de la page est réalisée.

De manière générale, pour une page Web donnée et un ensemble de souhaits définis nous repérons visuellement les éléments concernés dans la page, puis nous récupérons l'élément HTML correspondant et récupérons également la valeur de sa propriété qui est ciblée par le souhait. Chaque élément extrait associé à une propriété sera modélisé sous la forme d'une variable. Si on reprend l'exemple de la taille de police minimale, alors chaque élément textuel sera représenté par une variable dont le domaine sera les différentes tailles de polices possibles.

Un certain nombre de points est soumis à interprétation. Par exemple, si on s'intéresse à un petit article sur une page Web constitué d'un titre suivi de quelques paragraphes, il est évident que le titre est bien dissocié des paragraphes et qu'il a besoin d'un style différent. Mais qu'en est-il pour les paragraphes eux-mêmes ? Doit-on distinguer les paragraphes les uns des autres et les considérer comme des entités distinctes ayant leur propre style ? Ou bien doit-on considérer qu'il s'agit de la même chose et que leur apparence peut être liée à un style commun ? Voici le genre de questions que l'on peut se poser. Du fait d'une certaine interprétation, des choix sont donc réalisés par la personne réalisant l'extraction. Pour éviter les différences d'extraction, du fait de la réalisation manuelle, nous réalisons cette étape avec au moins deux personnes pour chaque page Web dans nos expérimentations actuelles.

Pour donner un exemple plus précis et concret nous utiliserons par la suite les trois souhaits (objectifs) définis dans le chapitre « Expérimentations » (chap. 8). Il s'agit des trois souhaits : « Contraste entre le texte et l'arrière-plan supérieur à x », « Brillance de couleurs d'arrière-plans similaire » et « Rester le plus proche possible des couleurs d'origine ». Dans le cas de ces trois souhaits, les propriétés ciblées sont des couleurs. Les variables créées à partir des éléments HTML ciblés par les préférences auront donc des domaines de valeur correspondant aux différentes couleurs pouvant être prises par les éléments HTML. Pour ce qui est de la préférence de proximité avec le contexte de couleur original de la page, la valeur de la couleur d'origine de l'élément sur la page doit aussi être récupérée. La figure 7.1 montre un exemple d'extraction réalisée sur une page Web du site

de la ville de Parempuyre. La variable 0 sera donc dédiée à la couleur de l'arrière-plan de l'entête gauche de la page Web. Sur le même principe, la variable 1 correspond à la couleur de l'arrière-plan général de la page. Davantage d'exemples sont disponibles en Annexe B.

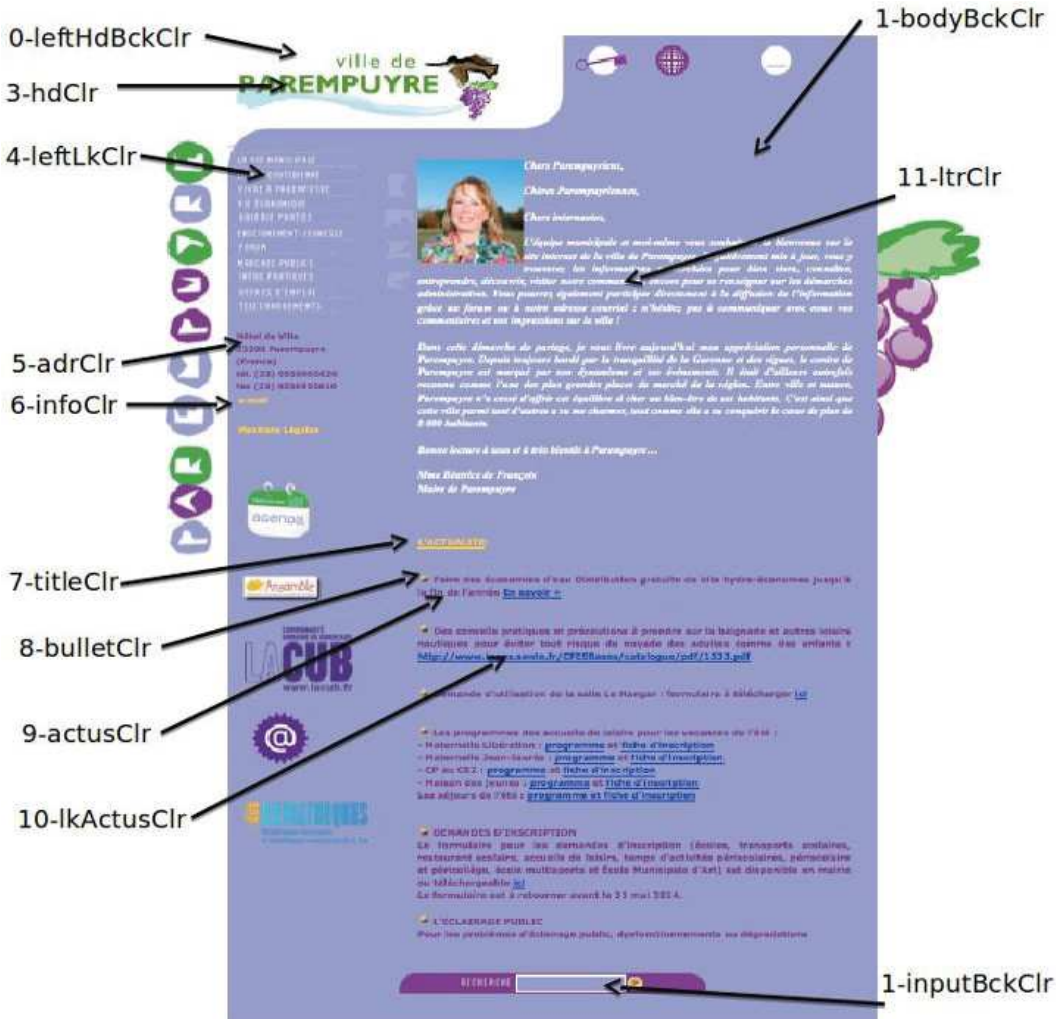


FIGURE 7.1 – Sélection des éléments sur la page Parempuyre

7.2 Données en entrée

Une fois que les différentes parties ainsi que les éléments de la page Web ciblés par les souhaits de l'utilisateur ont été repérés, on peut écrire de manière formelle l'ensemble des objectifs (formalisation d'un souhait dans le domaine de l'optimisation). Un format d'écriture pour les variables et les préférences a été défini afin d'écrire et de stocker les variables dans un fichier.

Le fichier issu de l'extraction manuelle des éléments contient deux types d'ensembles, le premier est celui des variables auxquelles on affecte la valeur actuelle dans la page. Pour notre exemple, utilisé dans la section précédente, chaque variable correspond à la couleur d'un élément, puisque tous les objectifs définis se basent sur la propriété de couleur. Chaque variable sera notée avec les informations suivantes : un numéro auto-incrémenté servant d'identifiant, un nom permettant d'indiquer de quel élément il s'agit sur la page et la valeur actuelle de la propriété associée. Nous stockons les valeurs de couleurs sous la forme de 3 octets. Chaque octet correspond à une composante de couleur rouge, verte ou bleue. Pour chaque composante, les valeurs seront donc comprises entre 0 (cette composante de couleur est nulle) et 255 (la saturation maximale pour cette composante de couleur). Ce qui nous donne 256 niveaux pour chaque composante soit 256^3 (environ 16.7 millions) couleurs. Si les objectifs avaient porté sur des tailles de polices alors on aurait des variables dont la valeur serait contenue dans le domaine des tailles de polices possibles.

0 - leftHdBckClr : 255 255 255
1 - bodyBckClr : 133 138 190
2 - inputBckClr : 133 138 190
3 - hdClr : 70 136 35
4 - leftLkClr : 255 255 255
5 - adrClr : 105 42 124
6 - infoClr : 248 192 57
etc.

FIGURE 7.2 – Exemples de variables pour le page Web Parempuyre

La définition des objectifs utilise les numéros d'identification des variables. La figure 7.3 donne un extrait des objectifs définis pour la page Web du site Parempuyre. Chaque ligne correspond à un objectif. « contrast », « distOrigClr » et « proxBrightness » sont trois fonctions permettant de calculer certaines propriétés sur la variable ou entre des variables. La fonction « contrast » donne le niveau de contraste de brillance entre les deux couleurs des deux variables indiquées. La valeur retournée est ensuite comparée à un seuil x fixé par l'utilisateur. Cette comparaison permet donc de connaître le niveau de satisfaction sur cette propriété de contraste. Pour ce qui est de la proximité de brillance entre les différents arrière-plans, la fonction « proxBrightness » évalue l'écart maximal entre les différentes variables (couleurs d'arrière-plans de la page) et cet écart est ensuite comparé à un seuil z défini par l'utilisateur, de la même manière que le seuil de contraste minimum. Les choses sont un peu différentes pour la distance de couleur (distance avec la couleur d'origine). Pour rappel il s'agit ici d'avoir la distance la plus faible afin de rester le plus proche possible de la couleur d'origine. La fonction « distOrigClr » ne dispose ici que d'un seul paramètre. Le calcul de la distance de couleur se fait entre la valeur de variable fournie et la couleur d'origine de cette variable qui a été renseignée lors de la déclaration de l'ensemble de variables. La distance est ensuite comparée au seuil y.

$\text{contrast}(0, 3) \geq x$
$\text{contrast}(1, 4) \geq x$
$\text{distOrigClr}(0) \leq y$
$\text{distOrigClr}(1) \leq y$
$\text{proxBrightness}(0,1,2) \leq z$
etc.

FIGURE 7.3 – Exemple d'objectifs pour le la page Web Parempuyre

Pour plus d'exemples, les fichiers d'entrée des différents sites Web utilisés pour les expérimentations (cf. Chapitre 8) sont disponibles en Annexe B. La section suivante donne le format de sortie du démonstrateur et l'application à la page Web.

7.3 Données en sortie et transformation de la page Web

Lorsque l'on sort de l'algorithme (un des critères d'arrêt est satisfait), le démonstrateur nous retourne un ensemble de variables. Il s'agit des variables que l'on avait dans notre fichier en entrée. Simplement elle sont affectées de la nouvelle valeur. La figure ci-contre (Figure 7.4) donne quelques exemples de retours possibles par le démonstrateur. Dans cet exemple, les deux premières variables sont des variables de type couleur, elle retournent par conséquent des couleurs. Des variables de natures différentes peuvent être utilisées mais le format de sortie reste le même (composé de paires var_id - valeur).

1 - 127 127 127
2 - 0 0 0
3 - 255 200 220
etc.

FIGURE 7.4 – Exemples de valeurs de retour associées aux variables

L'identifiant permet ensuite de retrouver l'élément HTML correspondant et de le modifier. Cette étape est également réalisée manuellement par modification du fichier HTML. L'étape d'extraction n'étant pas automatique, nous ne disposons pas de lien direct avec les éléments HTML de la page. L'automatisation de l'application de la modification dépend donc de l'automatisation de l'extraction des éléments HTML.

La section suivante décrit de manière générale le processus principal du démonstrateur et les liens avec les données d'entrée et de sortie.

7.4 Architecture générale

À la différence de l'extraction des données et de la transformation de la page, la recherche de la meilleure adaptation possible est réalisée de manière automatique à l'aide d'un algorithme génétique. Deux versions de NSGA ont été implémentées dans le même démonstrateur ce qui permet de comparer facilement les deux versions pour un même jeu de données en entrée. Le schéma 7.5 montre le fonctionnement général du démonstrateur et indique les données en entrée et en sortie.

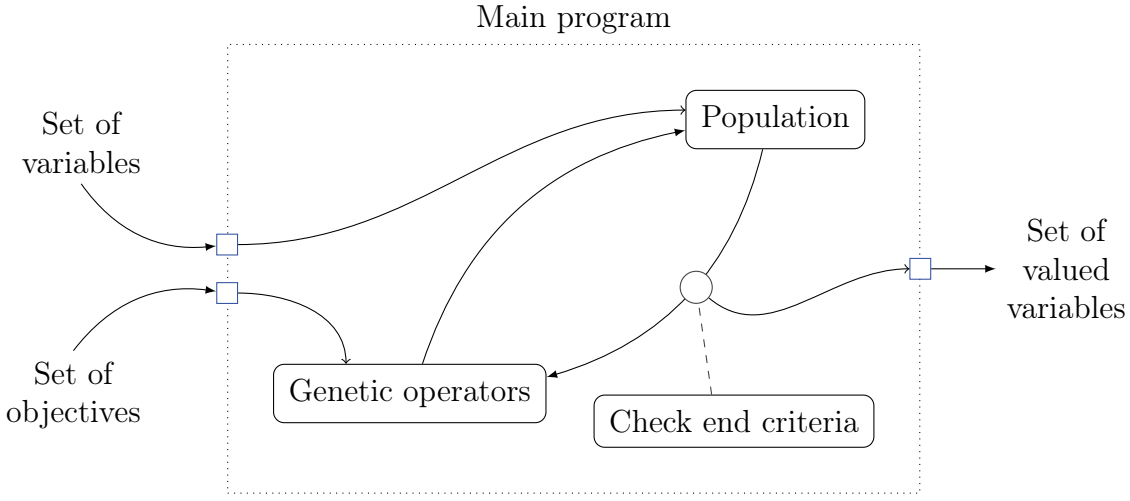


FIGURE 7.5 – Schéma résumé du démonstrateur

Le développement a été réalisé sous la forme d'une application en C++. L'architecture de cette dernière permet le développement facile de nouveaux types d'objectif qui permettront à l'utilisateur d'exprimer de nouveaux types de souhaits.

Les données sont entrées manuellement dans le programme et celui-ci s'exécute selon les paramètres définis. D'un côté nous avons le fonctionnement évolutionnaire où la population d'individus évolue à travers des opérateurs génétiques. D'un autre côté nous avons des critères d'arrêt dont la satisfaction est vérifiée régulièrement. L'évolution de la population se fait par la maximisation des scores des fonctions objectif. Pour un individu donné, plus la valeur retournée par la fonction objectif est importante et meilleur ce dernier sera. Lorsqu'un des critères d'arrêt est satisfait alors le démonstrateur retourne un ensemble de variables associées à leur nouvelle valeur.

7.5 Les différents paramètres

Divers paramètres sont disponibles afin de configurer le démonstrateur et les algorithmes qu'il utilise. Voici une liste des paramètres :

— **Algorithme** : L'algorithme à utiliser (NSGA-II ou NSGA-III)

— **Problème** : Le problème sur lequel on travaille (la page Web)

— **Objectifs** : Le ou les objectifs à utiliser

— **Taux de croisement** : Le taux de croisement vs mutation

— **Aggr** : Si l'exécution doit utiliser l'agrégation ou non

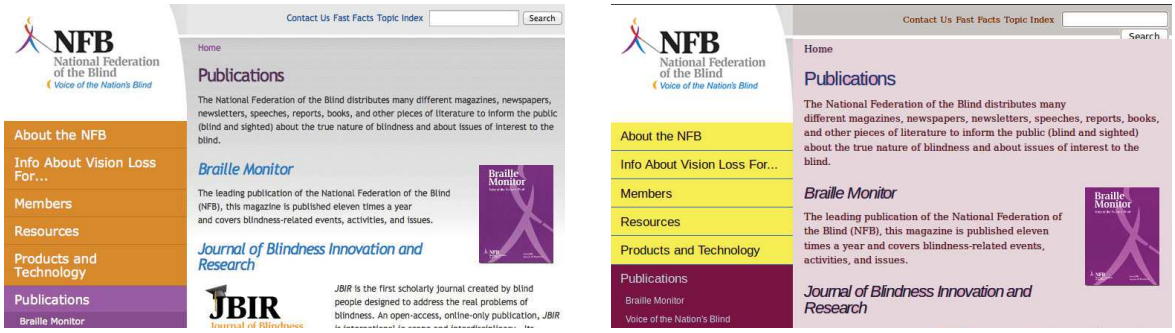
— **Aggr val** : Le nombre de fonctions objectif agrégées obtenues pour chaque type d'objectif

— **Taille population** : La taille de la population (NSGA-II)

— **Divisions** : Le nombre de divisions par fonction objectif (NSGA-III)

7.6 Exemples d'adaptations

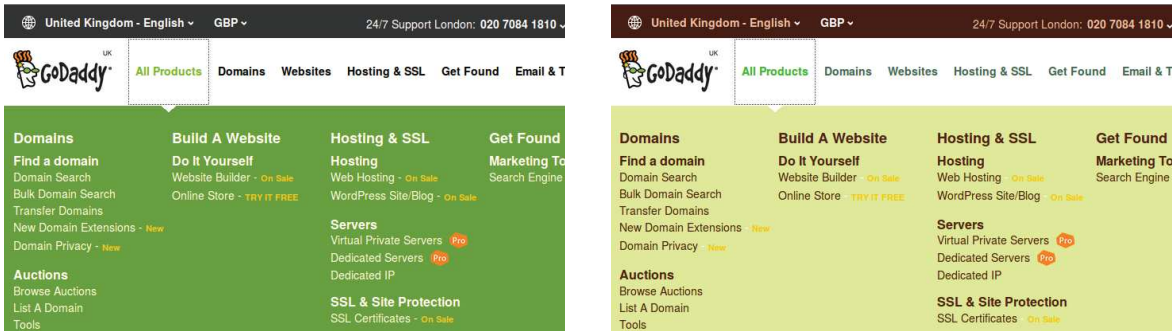
Voici des exemples d'adaptations calculées sur des pages réelles. Nous avons utilisé le site Web *Godaddy* et le site Web de la *NFB*. Pour ces deux exemples nous avons choisi d'appliquer deux objectifs à la fois. Le premier est d'avoir un contraste minimum de 30% entre le texte et l'arrière-plan et le second objectif concerne le fait de rester autant que possible proche des couleurs originales de la page. Ces deux objectifs combinés font sens, puisqu'on peut tout à fait vouloir conserver la page actuelle en augmentant simplement un peu les contrastes car le texte n'est pas assez lisible.



Page originale Page modifiée

FIGURE 7.6 – Transformation (contraste min et conserv. contexte couleur d'origine) - *NFB*

Sur le site de la « National Federation of the Blinds » (Figure 7.6), plusieurs éléments se trouvent impactés par la transformation. Le premier d'entre eux concerne le menu de navigation se trouvant sur la gauche. Ce menu offrait un contraste un peu faible entre le texte et l'arrière-plan. La nouvelle page offre un contraste bien meilleur sur ce point. On constate également que les nouvelles couleurs sont assez proches des anciennes du point de vue de leur teinte. La modification a en quelque sorte modifié le niveau de luminosité des couleurs afin de jouer sur le contraste. D'un autre côté, on observe également une transformation au niveau de l'arrière-plan du contenu principal. L'adaptation qui a été réalisée peut sembler un peu étrange mais elle est simplement due à la façon dont nous avons considéré les données de la page de départ. L'arrière-plan du contenu principal est réalisé par une image dégradée du gris vers le blanc. Pour passer outre cette image nous avons pris un gris intermédiaire se trouvant entre le plus foncé (en haut) et le plus clair du dégradé puis nous appliquons cette couleur sur tout l'arrière-plan à la place de l'image. De ce fait lorsque le calcul a été fait, le gris sélectionné a été considéré comme étant l'arrière-plan de l'ensemble du contenu principal de la page. On constate également un changement dans l'arrière-plan de la barre en haut de la page. La transformation réduit en partie le contraste entre le texte et cet arrière-plan. Cependant, c'est tout à fait normal que cela se produise, puisque l'on recherche une bonne adaptation en utilisant un algorithme initialisé aléatoirement. Le contraste résultant répond tout de même à l'objectif d'avoir au moins 30% de contraste.



Page originale Page modifiée

FIGURE 7.7 – Transformation (contraste min et conserv. contexte couleur d'origine) - *Godaddy*

Sur la page du site Web *Godaddy* (Figure 7.7), la modification a impacté plusieurs éléments. Tout d'abord ce que l'on remarque certainement le plus rapidement c'est le changement de couleur du menu déroulant. Au départ sur fond vert moyen avec du texte en blanc dessus, il se retrouve maintenant sur fond vert clair avec un texte noir. Depuis le menu original il existe deux grandes modifications pour répondre à la fois à l'objectif de contraste minimum mais aussi à l'objectif de conservation des couleurs originelles de la page. Soit on fonce le vert jusqu'à obtenir le contraste minimal souhaité avec le texte blanc. Soit on éclaircit le vert et à ce moment-là il faut basculer le blanc du texte vers du noir. Ici la seconde option a été prise par l'algorithme de résolution. Cette solution est tout à fait valable y compris concernant la conservation des couleurs d'origine même si le texte est noir au lieu d'être blanc. Cela vient de l'implémentation réalisée. Dans la fonction objectif permettant d'évaluer la proximité de deux couleurs, le noir et le blanc sont considérés comme proches dans le sens où ce ne sont pas vraiment des couleurs mais plutôt des niveaux de gris. La distance de couleur étant calculée en fonction de la différence de teinte entre les deux couleurs, le blanc et le noir étant tous les deux des nuances de gris, ils sont donc considérés comme proches. Une autre implémentation de cette fonction, en prenant par exemple en compte le niveau de saturation de la couleur, permettrait d'avoir des résultats différents.

Tout comme pour l'exemple précédent, certains contrastes ont été réduits, comme par exemple dans l'en-tête de la page. Cet effet est dû aux mêmes raisons que pour l'autre page Web. En effet, les couleurs ne sont pas choisies en fonction de la couleur d'origine mais plutôt dans le sens où on prend une couleur « au hasard » et on regarde si elle satisfait tous nos objectifs (ici 2 objectifs). Une couleur peut donc à la fois satisfaire l'objectif de contraste et l'objectif de proximité de teinte avec la couleur originelle sans pour autant être la même (même si la couleur initiale satisfaisait bien les deux objectifs).

7.7 Conclusion

Quelques étapes restent encore réalisées manuellement. Cependant la recherche d'une solution d'adaptation satisfaisant les souhaits de l'utilisateur est entièrement automatique. Ce démonstrateur implémente deux versions d'un algorithme évolutionnaire (algorithme génétique). Il permet donc la mise en œuvre d'expérimentations. Un certain nombre de paramètres sont disponibles afin de configurer le démonstrateur et notamment de paramétrer les algorithmes génétiques. Le chapitre d'expérimentations suivant (Chapitre 8) utilise ce démonstrateur et joue sur les différents paramètres afin de réaliser des comparaisons et de montrer dans quelle mesure la solution proposée résout le problème de l'adaptation d'une page Web dans un temps assez court.

Chapitre 8

Expérimentations

Préambule

Ce chapitre développe les résultats d'une expérimentation menée avec des algorithmes issus de deux grandes familles. Tout d'abord nous testons une approche issue de la théorie des préférences, qui propose des algorithmes exacts. Puis nous expérimentons le démonstrateur réalisé autour des deux versions (II et III) de l'algorithme génétique NSGA (cf. Chapitre 7). De nombreux résultats sont recueillis depuis des pages Web diverses et variées. Les résultats obtenus sur un site sont analysés en détails, puis une étude comparative plus large est menée sur une dizaine de sites Web. Une analyse du choix de certains paramètres est ensuite présentée.

Sommaire

8.1 Introduction **123**
8.2 Approches exactes **123**
8.3 Algorithmes génétiques **125**
8.3.1 Mise en œuvre 126
8.3.2 Corpus de pages Web réelles utilisé 131
8.3.3 Résultats 132
8.3.4 Impact du choix des paramètres 142
8.4 Conclusion **145**

8.1 Introduction

La mise en œuvre de divers algorithmes a permis la réalisation de nombreuses expérimentations. Celles-ci ont notamment été utiles dans les phases de détermination de la faisabilité et de la validation de l'approche globale.

Les algorithmes issus du domaine de la théorie des préférences, les algorithmes évolutionnaires ou génétiques, la modélisation sous la forme de problème de satisfaction de contraintes (CSP) etc. sont autant d'approches diverses permettant d'aborder le problème d'adaptation de pages Web. La nature variée des algorithmes (exact, déterministe ou non, stochastique, etc.) permet d'attaquer sous différents angles le problème.

Il faut rappeler que le nombre d'éléments que l'on traite est potentiellement très grand et que les domaines de valeurs peuvent également être très grands. La résolution n'est, par conséquent, pas triviale. Différentes méthodes doivent être abordées et adaptées au problème dans le but de les comparer en fonction de critères temporels ou de satisfaction. L'adaptation d'une page se faisant en « temps réel », c'est-à-dire à chaque chargement d'une nouvelle page au fil de la navigation de l'utilisateur, le temps de calcul se doit d'être le plus faible possible.

Ce chapitre développe les résultats d'expérimentations réalisées avec des algorithmes issus de deux grandes familles. Tout d'abord à partir de la théorie des préférences, qui propose des algorithmes exacts. Puis par la mise en œuvre de deux algorithmes évolutionnaires (génétiques) qui permettent de considérer le problème comme un problème d'optimisation.

8.2 Approches exactes

Des approches exactes comme celles issues de la théorie des préférences ont un certain nombre de limitations principalement liées à la taille de l'espace de recherche. Dans le contexte de l'adaptation de pages Web, le nombre d'éléments HTML ainsi que leurs caractéristiques physiques et leurs domaines de définition peuvent être très importants, rendant le nombre d'adaptations possibles lui aussi très grand. Cependant les préférences utilisateur sont le fondement de notre approche. Ces techniques de résolution méritent donc d'être étudiées plus en détail et de ne pas être directement écartées du fait de leurs limitations.

Plusieurs algorithmes ont donc fait l'objet d'implémentations. Les méthodes de résolution implicite ou explicite sont deux méthodes différentes permettant la représentation des adaptations possibles. Ces deux méthodes peuvent être utilisées avec toutes les sémantiques et ne changent pas le résultat. Puisque l'utilisation d'une méthode de résolution implicite n'apporte pas d'améliorations notables en terme de performance sur notre problème (cf. section 6.1) et que les deux méthodes font face aux mêmes limitations, nous avons utilisé la méthode de résolution explicite afin de simplifier le développement. Un petit nombre de sites Web ont été manuellement étudiés. Des ensembles de préférences ont aussi été définis et les éléments correspondant dans les pages de test ont été extraits. De plus, des ensembles de variables (éléments HMTL associés à une caractéristique) ont été aléatoirement générés créant ainsi quelques représentations de pages fictives.

Une des étapes clés communes à toutes les « sémantiques » d'algorithme est celle de la génération de l'ensemble comparatif (relations de préférence). L'algorithme makePrefRelationSet (Algorithme 14) décrit cette étape. La construction des relations de préférence se fait par la répartition des solutions de Ω dans les parties gauches et droites des relations de préférence. Cette répartition se base sur les formules (6.10) et (6.11).

Algorithme 14 Génération des relations de préférences (ensembles comparatifs)

```
function makePrefRelationSet(O, prefs)        ▷ O : outcome set, prefs : pref set
    comp = array();                           ▷ Initialize comparative set as an array
    for o = O.begin to O.end do
        for pref = prefs.begin to prefs.end do        ▷ Let pref = p ≻ q
            if o satisfies pref.p and not pref.q then
                add o to left set of comp[pref];
            else if o satisfies pref.q and not pref.p then
                add o to right set of comp[pref];
            end if
        end for
    end for
    return comp;
end function
```

L'équation 8.1 présente la complexité de cet algorithme 14 (génération de l'ensemble des relations de préférence). Elle peut s'exprimer en fonction du nombre d'atomes des formules logiques décrites dans le langage logique propositionnel $\mathcal{L}$ (voir section 6.1) et présentes dans l'ensemble de préférences $prefs$, ainsi que du nombre de variables présentes dans ces formules. Pour une formule P de $\mathcal{L}$, $Prop(P)$ indique le nombre d'atomes de P. v indique le nombre de variables (notées V_i, $i \in \{1, 2, ..v\}$) apparaissant dans les formules de $prefs$. On rappelle que $Dom(V_i)$ est le domaine de valeurs de la variable V_i. Le premier terme de l'équation 8.1 représente le nombre de solutions (outcomes) de O (chaque solution correspond à l'affectation d'une valeur pour chaque variable qu'il contient). Le second terme représente le nombre d'atomes qui doivent être évalués pour une solution.

$$\left(\prod_{i=1}^{v} |Dom(V_i)|\right) \times \left(\sum_{P_j \in prefs} Prop(P_j)\right) \tag{8.1}$$

Table 8.1 – Algorithmes prefs - Quantifications et estimation du temps d'exécution

	Google	Ex_{3-6}	Ex_{4-10}	Facebook	*Google+*	CNN
Objects	2	3	4	9	11	74
Variables	3	6	10	30	50	250
Preferences	3	6	10	30	50	250
Domains Card	6	6	6	6	6	6
Outcomes	216	46656	60466176	6^{30}	6^{50}	6^{250}
Time	$<$ 1ms	250ms	6mn	$>$ 24h	$>$ 24h	$>$ 24h

Les temps de calcul de notre implémentation de l'algorithme 14 montrent sans surprise que l'utilisation de cette approche n'est pertinente que sur de très petits ensembles de variables à faible domaine de valeurs (cf. Table 8.1). Dès que l'on traite des pages Web réelles, et donc bien plus conséquentes ici (en dehors de Google), il n'est plus possible de trouver une solution dans un temps raisonnable (quelques secondes).

La page de Google donne de bons résultats car elle très petite. Il s'agit de la page de recherche très simple et non pas de la page de résultats de recherche. Cette page de recherche ne contient que très peu d'éléments importants et pertinents concernant les préférences définies. Les préférences utilisées ici sont des préférences de type taille de police avec un domaine de définition de 6 valeurs (6 tailles de police différentes). Les variables sont donc ici les objets associés à la taille de la police. Le nombre de préférences utilisateur reste inchangé pour tous les exemples. Il en va de même pour le domaine des variables, seul le nombre de variables (qui dépend de la page) change et par conséquent l'espace de recherche. Le temps annoncé pour les exemples les plus complexes ne signifie pas qu'avec une machine plus puissante on pourrait descendre significativement plus bas. En réalité d'après les estimations, plusieurs milliers d'années seraient nécessaires afin de réaliser le calcul pour la page d'inscription de Facebook et un temps encore supérieur pour les autres exemples. De ce fait il n'est même pas possible d'envisager une telle approche ne serait-ce que pour faire des pré-calculs d'adaptation en arrière-plan.

Les sémantiques d'algorithmes permettent entre autres d'assouplir ou de durcir le respect des préférences. Cette notion de sémantique peut être intéressante dans le sens où elle permet d'ordonner de diverses manières les solutions en fonction par exemple du type de préférences données. Cependant l'ensemble de préférences définies par l'utilisateur ne doit pas contenir de conflits. C'est-à-dire que les dépendances entre les préférences ne doivent pas entraîner d'impossibilités de trouver une solution. Quelle que soit la sémantique, si les préférences sont contradictoires, il sera impossible d'ordonner les solutions.

D'autres approches comme les algorithmes génétiques n'ont pas cette notion de sémantique a priori. Cependant il est possible de donner un poids à chaque préférence afin de lui donner plus ou moins d'importance. De plus les conflits entre préférences seront résolus, par compromis, et de manière approchée puisqu'il s'agit de réaliser une optimisation.

8.3 Algorithmes génétiques

Cette approche est celle qui a été la plus approfondie et qui a fait l'objet d'un plus grand nombre d'expérimentations. Elle est présentée en partie dans les articles [BHM14a] et [BHM15]. Ici le temps d'adaptation maximal, qui est un paramètre très important pour l'utilisateur final, a été arbitrairement fixé à 10 secondes. Ce temps est bien entendu trop important dans le cas d'un usage en conditions réelles. Si 10 secondes doivent être ajoutées à chaque chargement de page alors l'outil n'est pas viable, même si les résultats sont bons et permettent d'améliorer le confort de lecture. Cependant ce seuil a été fixé pour déterminer si dans certains cas, passer quelque secondes supplémentaires à attendre ne pourrait pas donner de bien meilleurs résultats. Dans l'idéal, on va plutôt chercher à obtenir des temps de l'ordre de la seconde et tirer les meilleurs résultats possibles pour ces quelques secondes..

Il est possible de jouer sur un grand nombre de paramètres lorsque l'on utilise les algorithmes NSGA-II ou NSGA-III. Ces paramètres ont chacun leur impact sur la vitesse de convergence ce qui influe donc sur le temps de calcul, le nombre de générations et la qualité des solutions retournées. En effet, la taille de population est une composante majeure de la complexité algorithmique, puisque dans NSGA-II et NSGA-III cette population est affecté d'une puissance de 2 (MN^2 ou N est la taille de la population et$M$ le nombre de fonctions objectif). Par conséquent, la valeur choisie aura un impact notable sur l'exécution de l'algorithme. Il en va de même pour la probabilité de croisement et de mutation, qui définit la part d'aléatoire qui intervient à chaque génération. On a également des paramètres qui sont propres à une variante de NSGA. Par exemple, dans NSGA-III il est possible de choisir le nombre de points de références qui conditionnent le niveau de précision de l'opérateur de sélection basée sur la diversité. Cette souplesse offerte par les nombreux paramètres disponibles nous permet d'ajuster assez précisément plusieurs grandeurs importantes afin d'ajuster le comportement des algorithmes. Cependant cette souplesse n'est pas forcément un avantage. En effet, les paramètres ne peuvent pas être pris au hasard et il n'est pas évident que les meilleurs résultats pour divers types de problèmes soient obtenus avec les mêmes paramètres.

8.3.1 Mise en œuvre

Cette section détaille l'ensemble de la configuration pour les expérimentations menées avec les algorithmes génétiques NSGA-II et NSGA-III. Comme il a été mis en avant dans les sections précédentes, l'apport de cet algorithme génétique (NSGA) est son approche par optimisation. De plus il s'agit d'un algorithme multi-objectifs ce qui permet de prendre en compte plusieurs objectifs sans les agréger en un seul et de ce fait perdre toutes les informations sur la satisfaction de chacun des objectifs lors de la résolution. La faisabilité de cette approche a été montrée dans l'article [BHM14b] et une expérimentation plus large a été réalisée dans l'article [BHM14a].

Différentes valeurs sont utilisées pour chaque paramètre. Nous appellerons ici configuration une valuation de tous les paramètres. En d'autres termes une configuration est un ensemble de valeurs, une par paramètre. Le tableau 8.2 décrit les configurations et paramètres communs qui seront utilisés pour NSGA-II et NSGA-III. Toutes les configurations possibles ne sont pas proposées ici car certaines d'entre elles ne seraient pas cohérentes. D'autres seront détaillées dans les sections suivantes et feront l'objet d'une analyse particulière. Par exemple nous ne faisons pas entrer en jeu la probabilité de croisement et mutation comme un paramètre avec plusieurs valeurs possibles. Celle-ci a été définie et fixée à l'avance à 0.94 (probabilité de croisement sans mutation), suite à une étude rappor-

TABLE 8.2 – Paramètres communs de configuration pour NSGA-II et NSGA-III

Paramètre	**Valeur(s)**
Probabilité de croisement/mutation	94%/6%
Temps d'exécution maximum	10s
Nombre de générations maximum	sans limite
Ensembles d'objectifs	4
Exécutions par configuration	60 répétitions

tée en section 8.3.4. D'autres paramètres seront analysés plus en détail comme le nombre de points de référence utilisés (NSGA-III) ou encore la taille de la population.

TABLE 8.3 – Paramètre de configuration propre à NSGA-II

Paramètre	**valeur(s)**
Taille de population	{100, 150, 200, 250, 300}

TABLE 8.4 – Paramètres de configuration propres à NSGA-III

Paramètre	**valeur(s)**
Taille de population minimale	256
Nombre de divisions par fct. objectif	De 2 à p

De petites variations dans le paramétrage entre NSGA-II et NSGA-III existant, les configurations de ces deux algorithmes divergent donc légèrement sur certains points. Quand NSGA-II possède comme paramètre la taille de la population, dans NSGA-III cette taille de population est quant à elle déterminée en fonction à la fois du nombre de fonctions objectif mais également du nombre choisi de divisions de chaque fonction objectif. Le nombre de divisions fait donc également partie des paramètres propres à la variante NSGA-III. Le tableau 8.3 ci-contre, donne le paramétrage de la population pour NSGA-II et le tableau 8.4 donne le paramétrage pour NSGA-III.

La part d'aléatoire dans les algorithmes génétiques étant non négligeable, le nombre d'exécutions pour chaque configuration doit être important. Le nombre d'exécution a donc été fixé à 60. Cette valeur assez importante permet de lisser l'effet que provoque l'aléatoire. Ainsi nous manipulerons des moyennes sur les temps d'exécutions, le nombre de générations ou le taux de satisfaction. En plus de ces moyennes, les valeurs minimales et maximales parmi les 60 exécutions sont conservées ainsi que l'écart-type. Ces statistiques vont notamment permettre la comparaison entre les différentes configurations des algorithmes.

La population, paramètre ayant une forte implication dans la complexité de ces deux algorithmes génétiques, apparaît bien souvent dans la littérature avec des valeurs proches de 200 individus. Dans cette expérimentation nous étendons ce paramètre à des valeurs comprises entre 100 et 300 par pas de 50 individus. Ces cinq tailles différentes de population auront pour but d'analyser le comportement de l'algorithme sur nos problèmes spécifiques d'adaptation et ainsi de déterminer la zone où on obtient les meilleurs résultats. À cause de la complexité algorithmique en MN^2 des deux variantes NSGA-II et NSGA-III, où M est le nombre de fonctions objectif et N la taille de la population, la taille de la population se doit de rester limitée. À l'opposé, la taille de population ne peut diminuer en deçà d'un certain seuil. La réduction de la taille de la population peut porter préjudice lors de la génération de la descendance en limitant les interactions possibles. Il s'agira alors de trouver le bon compromis taille de la population versus nombre de générations, puisque le nombre de générations dépend directement de la taille de la population (pour un nombre de fonctions objectif fixé). Les différentes tailles de population ont été expérimentées dans la référence [BHM14a] et sont discutées dans la section 8.3.4. Dans la section 8.3.3, la taille de la population a été fixée à 250 qui est un bon compromis pour le temps d'exécution et le nombre de générations (voir section 8.3.4).

Ces deux variantes d'algorithme sont non déterministes et incluent donc une part d'aléatoire non nulle. Les opérateurs génétiques de croisement, de mutation et de sélection tels que définis, ne permettent pas d'assurer la convergence. Dans certains cas, il est possible de stagner dans une portion de l'espace de recherche autour d'un optimum local. Lorsque cela se produit, il y a très peu de chances pour qu'une bonne solution soit retournée dans un temps acceptable. De ce fait, pour éviter de passer trop de temps à chercher une bonne solution sans succès, le temps d'exécution est limité à 10 secondes. À la fin de ce temps imparti, l'exécution s'arrête et retourne la population courante. Cette population contient entre autres les meilleurs individus trouvés compte tenu du temps écoulé (10s)

NSGA nous permet entre autres de trier nos solutions en fronts de non-domination en fonction de plusieurs fonctions objectif. Dans cette expérimentation nous définissons trois types de préférences générales, que nous appellerons plutôt objectifs généraux dans le contexte de NSGA. Les objectifs représentent la version opérationnelle de nos préférences. Ces derniers couvrent de nombreux problèmes rencontrés par les personnes ayant une basse vision. GP_1, GP_2 et GP_3 sont ces trois objectifs généraux.

— GP_1 : Brillance d'arrière-plan uniforme
— GP_2 : Contraste minimum entre le texte et son arrière-plan direct
— GP_3 : Rester autant que possible proche des couleurs d'origine

Lorsque ces objectifs sont associés à un site web, alors ils vont s'instancier sur tous les éléments les concernant afin de donner un ensemble d'objectifs et de fonctions objectif associées. Cet ensemble d'objectifs est donc spécifique à un site web. Pour le site web *Parempuyre* qui fait partie de notre expérimentation, nous aurons trois variables pour représenter nos trois arrière-plans. La variable *inputBckClr* (input background color) correspond à l'arrière-plan des champs de saisie, la variable $leftHdBckClr$ (left header background color) concerne quant à elle l'arrière-plan du menu de gauche, et l'arrière-plan du corps de la page est représenté par la variable *bodyBckClr* (body background color). Nous aurons également des variables pour la couleur des textes comme $leftLkClr$ (left link color) pour la couleur des liens du menu de gauche et *hdClr* (text header color) concernant la couleur du texte dans l'entête. Lorsque l'on applique l'ensemble d'objectifs généraux sur ce site web particulier (sur cet ensemble de variables) on obtient un ensemble d'objectifs dont voici un extrait représentatif :

$$
\begin{aligned}
P_1 &= \{proxBrightness(leftHdBckClr, bodyBckClr, inputBckClr)\} \\
P_2 &= \{contrast(leftHdBckClr, hdClr) >= 30, \\
&\qquad contrast(bodyBckClr, leftLkClr) >= 30, ...\} \\
P_3 &= \{distOrigClr(leftHdBckClr) < 40, \\
&\qquad distOrigClr(bodyBckClr) < 40, ...\}
\end{aligned}
$$

TABLE 8.5 – Configuration des objectifs utilisateur

Paramètre	Valeur(s)
Ensembles d'objectifs	4
Différence maximale de brillance	40%
Distance maximale de couleur	40°
Contraste minimum	30%

Le fait d'avoir des arrière-plans ayant une brillance similaire permet de rendre plus confortable la lecture. En effet, la perception de l'œil est relative et ce dernier a besoin de temps pour s'adapter à une nouvelle luminosité. Cet objectif global ($P1$) est défini pour toutes les variables d'arrière-plan de la page. Cet objectif est implémenté sous la forme d'une seule fonction objectif qui prend en paramètre l'ensemble des variables d'arrière-plan de la page. Cette fonction est globale sur tous les arrière-plans mais il est possible de la décomposer en une série de fonctions binaires. Dans ce cas, il suffit de lier tous les couples possibles de variables de manière à former une clique. Dans notre cas, nous n'utiliserons qu'une seule fonction objectif comme défini par l'algorithme 10.

Concernant les deux autres objectifs (P_2 et P_3), ceux-ci sont aussi basés sur la propriété de couleur. P_2 est l'ensemble d'objectifs renseignant un contraste minimum entre le texte et son arrière-plan direct. Cet objectif permet notamment d'améliorer la lisibilité. Ici nous parlons d'un contraste de brillance, c'est-à-dire que l'on va comparer les valeurs de brillance (cf. section 2.4.2) des deux éléments (le texte et l'arrière-plan). La brillance sur laquelle cet objectif se base (comme tous les autres objectifs utilisant la brillance) utilisent la brillance (ou luminance perçue) telle que définie par le W3C (voir section 2.4.2). L'implémentation de la fonction de contraste de brillance minimal est celle de l'algorithme 11.

Le dernier ensemble d'objectifs P_3 diffère légèrement des deux autres. Dans ce dernier cas, les fonctions objectif qui en résultent porteront sur un seul élément de la page (variable) qui sera associé à une constante. Cette constante représentera la couleur d'origine de l'élément à laquelle sa couleur actuelle est comparée. Cet objectif permet la conservation du contexte de couleur originel d'éléments de la page. L'algorithme 13 décrit l'implémentation utilisée dans cette expérimentation.

Certains de ces types d'objectifs peuvent avoir un intérêt à être manipulés tout seuls. À l'inverse, d'autres ne seraient pas pertinents, comme par exemple le fait de rester le plus proche possible du contexte de couleur original. Si cet objectif est utilisé tout seul alors il ne sert à rien de calculer quoi que ce soit, il suffit de garder la page d'origine pour tout satisfaire. Cet objectif devra par conséquent être combiné à un ou plusieurs autres. C'est ce qui se passe généralement dans la pratique puisque rares seront les cas où un seul type d'objectif sera utilisé à la fois. La combinaison de plusieurs types d'objectifs est donc étudiée ici. Les S_i représentent les combinaisons de types d'objectif. Les ensembles qui sont utilisés dans cette expérimentation sont les suivants :

$$S_1 = P_2 \tag{8.2}$$
$$S_2 = P_2 \cup P_3 \tag{8.3}$$
$$S_3 = P_1 \cup P_2 \tag{8.4}$$
$$S_4 = P_1 \cup P_2 \cup P_3 \tag{8.5}$$

Le contraste minimal de brillance de couleur est la base de nos ensembles d'objectifs. Il apparaît dans toutes les combinaisons utilisées. Dans l'ensemble S_1, on ne trouve que l'objectif de contraste minimal de brillance de couleurs. Dans S_2 il est associé à la conservation du contexte de couleur original de la page. La constitution de l'ensemble se fait par union des objectifs de contraste (P_1) et des objectifs consistant à préserver des couleurs proches de l'original (P_3). Sur le même principe, deux autres ensembles sont générés. S_3 associe le contraste minimal avec l'uniformité des brillances d'arrière-plan. Dans cette configuration, une dépendance apparaît clairement entre ces deux objectifs puisque, que ce soit le contraste minimal de brillance de couleur ou l'uniformité des brillances d'arrière-plan, les deux types d'objectif portent sur la brillance d'éléments en commun. Dans le cas de S_4, on associe le contraste minimal avec la conservation de contexte original de couleurs mais aussi avec l'uniformité des brillances d'arrière-plan. En d'autres termes, S_4 combine les trois types d'objectif définis.

Tous les ensembles d'objectifs définis ici seront implémentés (pour une page donnée) de deux manières différentes. La première consiste à transformer chaque objectif de l'ensemble en une fonction objectif. De cette manière, on aura un score pour chaque objectif et le tri des individus tiendra compte de chacun d'entre eux individuellement. La seconde implémentation consistera à utiliser l'agrégation d'objectif. C'est-à-dire que l'on implémentera plusieurs objectifs dans une seule fonction objectif. Le but étant de réduire le nombre de fonctions objectif dans l'algorithme qui occupe une part importante de temps de calcul. Lorsque plusieurs objectifs sont agrégés dans une seule fonction objectif, alors tous les objectifs sont évalués mais le tri des individus ne se fera qu'en fonction du score total (celui de la fonction objectif agrégeant les objectifs). Ainsi, il y a moins de fonctions objectif et par conséquent le tri se fera avec moins de valeurs. Le modèle d'agrégation choisi consiste à regrouper les objectifs en fonction de leur type. Autrement dit chaque fonction objectif agrégée n'aura qu'un seul type d'objectif (par exemple contraste de brillance minimum). Pour chaque groupe d'objectifs (groupement par type) on utilise deux fonctions objectif qui se répartissent aléatoirement, mais équitablement, les objectifs de ce groupe. Trois types d'objectifs sont définis pour cette expérimentation. Dans nos configurations, nous n'aurons qu'une seule occurrence de l'objectif qui concerne l'uniformité des brillances d'arrière-plan portant sur tous les arrière-plans de la page. Par conséquent pour ce type d'objectif on n'aura qu'une seule fonction objectif. Le nombre de fonctions objectif agrégées pourra donc varier de 2 à 5 selon les types d'objectifs utilisés.

La section suivante donne les résultats de l'exécution des deux algorithmes NSGA-II et NSGA-III sur des pages réelles. Ces deux algorithmes possèdent chacun un bon nombre de paramètres, les sections suivantes donnent également un aperçu de l'impact du choix de valeur pour ces paramètres.

8.3.2 Corpus de pages Web réelles utilisé

Des pages Web diverses ont été sélectionnées à la fois pour leur diversité en terme de nombre d'éléments, par rapport au nombre de couleurs utilisées mais aussi parce que leur apparence est propice à des problèmes d'accessibilité. Des sites Web de contenu, des sites de formation ou bien des plate-formes de e-commerce ont donc été utilisés. Une partie de ces pages Web (Legibase, BLMyAccount et BLFamilyPortal) ont été étudiées dans le cadre d'un projet avec l'entreprise Berger Levrault qui travaille sur l'accessibilité de ses applications. La page Legibase est en ligne ; les pages BLMyAccount et BLFamilyPortal faisaient partie d'un projet en cours de développement. Elles ont servi de support à un échange avec l'équipe qui les développait. L'ensemble des pages utilisées se trouve en Annexe A avec leurs caractéristiques. En Annexe B vous trouverez les illustrations concernant l'extraction des variables depuis les pages (extraction réalisée manuellement) et le contenu des fichiers permettant de déclarer la liste des variables et l'ensemble d'objectifs sur ces dernières.

Après avoir défini les groupes d'objectifs, il faut définir les variables sur lesquelles ces objectifs vont porter. Étant donné que la phase d'extraction et d'annotation automatique (reconnaissance) des éléments HTML dans les pages est en cours d'élaboration, les données doivent donc être extraites à la main. Cette extraction a été réalisée par au moins deux personnes afin de limiter la subjectivité et pour obtenir une certaine cohérence entre les sites. Le nombre de variables varie pour nos exemples d'une dizaine à une centaine. L'ensemble de ces variables sont des variables de type couleur. La caractéristique de couleur est une de celles qui a le plus grand domaine de définition et sur laquelle plusieurs types d'objectifs peuvent être définis. Nous avons fixé le nombre de couleurs à 2^{15}, soit 32 768 couleurs. Les couleurs sont représentées sous la forme de trois composantes de couleurs R, V et B (Rouge, Vert et Bleu). Chacune de ces trois composantes a un domaine de définition de 32 valeurs. Habituellement, dans le modèle de représentation RVB chacune des composantes possède 256 valeurs ce qui donne environ 16.7 millions de couleurs (256^3). Même si on utilise ici des algorithmes d'optimisation qui ne génèrent pas toutes les solutions, pour limiter l'ampleur de l'espace de recherche nous avons réduit ces domaines. La réduction du domaine de définition aurait pu être plus importante. Par exemple 4 (resp. 6) valeurs par composante de couleurs nous auraient donné 64 (resp. 216) couleurs par variable. Cependant, plus l'espace de recherche est réduit (réduction du domaine de définition des couleurs), plus on réduit la diversité des solutions et plus on risque de supprimer les bonnes solutions.

8.3.3 Résultats

Deux algorithmes génétiques ont donc été implémentés et testés, NSGA-II et NSGA-III (Non-dominated Sorting Genetic Algorithm). Des différences notables ont pu être observées entre ces deux versions en fonction des données. Dans un premier temps, nous présenterons les résultats détaillés des deux algorithmes sur le site *Godaddy*, puis nous proposerons une étude plus large (sur une dizaine de sites Web). L'ensemble des résultats a été obtenu sur un ordinateur portable grand public disposant d'un processeur quatre cœurs (2.6Ghz) et muni de 8GB de RAM DDR3.

NSGA-II sur une page du site *Godaddy*

La table 8.6 montre les résultats d'exécution de NSGA-II pour les ensembles d'objectifs S_1, S_2, S_3 et S_4 sur une page de la plate-forme d'e-commerce *Godaddy*. Dans cette première table, les ensembles d'objectifs utilisés ne sont pas agrégés : chaque objectif est implémenté par une fonction objectif. Pour l'ensemble d'objectifs S_1, le nombre d'objectifs et donc le nombre de fonctions objectif est de 14. Ce nombre atteint 37 pour l'ensemble le plus complexe S_4. Le nombre de variables extraites est de 22. C'est-à-dire que les objectifs définis portent sur 22 éléments HTML associés à leur caractéristique de couleur pour cette page Web. Le tableau de résultats est trié en fonction du nombre de fonctions objectif de chaque ensemble d'objectifs, d'où l'inversion des colonnes présentant les résultats de S_2 et de S_3 qui est volontaire. Ce tri se base sur l'hypothèse que pour une page donnée, plus le nombre de fonctions objectif est grand et plus le problème est complexe. Les temps d'exécution se situent autour de 6 secondes pour l'ensemble d'objectifs le plus simple (S_1) et montent jusqu'à 10 secondes, qui sont dans cette expérimentation la borne maximale fixée pour le temps d'exécution. On constate pour ce premier ensemble d'objectifs, qui n'a que 4% d'exécutions atteignant les 10s, que l'écart type pour le temps d'exécution est d'environ 1.6s. On constate donc, que lorsqu'une exécution trouve une bonne solution (solution qui satisfait tous les objectifs) alors son temps de calcul est assez stable. Alors que S_1 regroupe

TABLE 8.6 – NSGA-II sur « Godaddy » **sans** agrégation des ensembles d'objectifs

Ens. objs.	S_1	S_3	S_2	S_4
Fcts. obj.	14	15	36	37
Temps exec. (s)				
Moyenne	5.87	9.89	10.00	10.00
Écart type	1.64	0.53	0.00	0.00
Min	3.67	6.64	10.00	10.00
Max	10.00	10.00	10.00	10.00
Générations				
Moyenne	244	376	200	193
Écart type	64.8	27.6	7.9	6.4
Min	154	264	192	183
Max	397	415	228	224
Exec. $\geq$ 10s				
Pourcentage	4	94	100	100
Objs. satisfaits	9.5	10.4	7.2	7.9
Écart type	0.5	3.8	1.3	1.4
Min	9	3	4	5
Max	10	14	10	12

tous les objectifs de contraste minimal, S_3 quant à lui vient ajouter l'objectif d'uniformité des brillances d'arrière-plan. Avec S_3, le nombre d'exécutions de l'algorithme qui amènent à une bonne solution descend de manière importante pour atteindre 6% (94% d'exécutions non terminées). Cependant, on trouve toujours des bonnes solutions pour un certain nombre d'exécutions.

Les deux ensembles d'objectifs S_2 et S_4 ne permettent pas de trouver de bonnes solutions en moins de 10 secondes. De plus, le nombre de fonctions objectif satisfaites, pour la moyenne de ces exécutions non terminées, n'excède pas 8 et dans le meilleur des cas le nombre maximum de fonctions objectif satisfaites n'atteint pas la moitié du nombre de fonctions objectif total. Pour ces deux derniers ensembles, le nombre de fonctions objectif est bien plus important que pour les deux autres. En plus de cela, le nombre de dépendances entre ces fonctions objectif est fortement augmenté puisqu'elles portent toujours sur les mêmes 22 variables.

L'agrégation des objectifs dans une seule fonction objectif est utilisée dans les algorithmes simple-objectif. Néanmoins, même si nous expérimentons ici des algorithmes multi-objectifs, il peut être intéressant de tester l'effet de l'agrégation. On profite d'avoir des algorithmes multi-objectifs pour ne pas agréger en une seule fonction objectif mais en plusieurs fonctions objectif (cf. Section 8.3.1). La table 8.7 regroupe les résultats pour les quatre ensembles d'objectifs S_{1agr}, S_{2agr}, S_{3agr} et S_{4agr}. Cette fois-ci les temps d'exécution moyens commencent à environ 0.4 secondes (S_{1agr}). De plus, l'intégralité des exécutions amène à une solution satisfaisant tous les objectifs. Le temps d'exécution est faible mais surtout très stable avec un minimum et un maximum proches de la moyenne et un écart type réduit. On retrouve un comportement similaire pour l'ensemble d'objectifs S_{3agr} pour lequel toutes les exécutions donnent une bonne solution en un temps assez court. Cependant pour les deux derniers ensembles d'objectifs, il n'y a aucune amélioration notable. Sur cette page, les bénéfices de l'agrégation d'objectifs sont donc présents mais limités. Les deux cas les plus complexes (S_{2agr} et S_{4agr}) restent trop difficiles. On notera que le nombre de générations pour un temps donné a augmenté, ce qui est justifié par la complexité de l'algorithme où le nombre de fonctions objectif entre en jeu.

TABLE 8.7 – NSGA-II « Godaddy » avec ensembles d'objectifs agrégés

Ens. objs.	S_{1agr}	S_{3agr}	S_{2agr}	S_{4agr}
Fcts. obj	2	3	4	5
Temps exec. (s)				
Moyenne	0.41	0.46	10.00	10.00
Écart type	0.05	0.10	0.00	0.00
Min	0.32	0.32	10.00	10.00
Max	0.49	0.84	10.00	10.00
Générations				
Moyenne	21	22	414	386
Écart type	2.5	5.1	2.4	1.9
Min	17	16	409	383
Max	26	41	418	390
Exec $\geq$ 10s.				
Pourcentage	0	0	100	100
Objs. satisfaits	–	–	0.6	1.3
Écart type	–	–	0.8	0.4
Min	–	–	0	1
Max	–	–	2	2

NSGA-III sur une page du site *Godaddy*

TABLE 8.8 – NSGA-III « Godaddy » avec **sans** agrégation des ensembles d'objectifs

Ens. objs.	S_1	S_3	S_2	S_4
Taille pop.	256	256	668	704
Fcts. obj.	14	15	36	37
Temps exec. (s)				
Moyenne	2.71	3.53	10.00	10.00
Écart type	0.48	0.70	0.00	0.00
Min	1.76	2.43	10.00	10.00
Max	3.63	5.78	10.00	10.00
Générations				
Moyenne	56	65	12	10
Écart type	9.8	12.7	0.0	0.2
Min	37	45	12	10
Max	75	106	12	11
Exec. $\geq$ 10s				
Pourcentage	0	0	100	100
Objs. satisfaits	–	–	15.5	15.3
Écart type	–	–	1.2	1.2
Min	–	–	13	13
Max	–	–	18	18

NSGA-III a été conçu afin de prendre en charge un plus grand nombre de fonctions objectif. En effet une des limitations de NSGA-II était sa chute de performance lorsque le nombre de fonctions objectif grandissait. Les résultats de l'exécution de cette version de NSGA sur les quatre ensembles montrent une amélioration globale notable (cf. Table 8.8). Sans agrégation, les exécutions pour les deux ensembles d'objectifs S_1 et S_3 retournent toutes des solutions satisfaisant l'intégralité des fonctions objectif. Le nombre de générations est bien moins élevé que la même configuration avec NSGA-II. Le temps passé par génération est plus long, mais l'algorithme semble converger plus rapidement. De plus, lorsque l'on arrive sur les deux cas les plus complexes (S_2 et S_4) même s'il n'est toujours pas possible d'obtenir de bonnes solutions, le niveau de satisfaction des solutions obtenues est doublé. Alors que l'on tourne autour des 20% de fonctions objectif satisfaites avec NSGA-II, on passe autour des 40% avec NSGA-III. Le nombre de générations réalisées pour les deux derniers ensembles d'objectifs est très bas. Le nombre de fonctions objectif complexifie grandement les calculs relatifs aux points de références. Ce nombre de générations très bas peut en partie justifier le fait qu'aucune bonne solution ne soit trouvée dans les 10 secondes. Un nombre plus grand pourrait permettre de converger.

TABLE 8.9 – NSGA-III « Godaddy » avec agrégation des ensembles d'objectifs

Ens. objs.	S_{1agr}	S_{3agr}	S_{2agr}	S_{4agr}
Taille pop.	256	256	256	256
Fcts. obj.	2	3	4	5
Temps exec. (s)				
Moyenne	0.38	0.40	9.11	10.00
Écart type	0.04	0.04	2.68	0.00
Min	0.27	0.31	0.94	10.00
Max	0.46	0.48	10.00	10.00
Générations				
Moyenne	21	21	382	392
Écart type	2.3	1.9	111.6	3.4
Min	16	17	43	385
Max	26	25	432	400
Exec $\geq$ 10s				
Pourcentage	0	0	90	100
Objs. satisfaits	–	–	2.5	3.0
Écart type	–	–	0.6	0.7
Min	–	–	1	2
Max	–	–	3	4

Même si NSGA-III est prévu pour manipuler beaucoup de fonctions objectifs, l'agrégation de fonctions objectif reste tout de même potentiellement un plus puisque le nombre de fonctions objectif reste toujours au cœur de la complexité algorithmique. Avec l'agrégation des objectifs en quelques fonctions objectif agrégées, le comportement global de l'algorithme sur l'exemple *Godaddy* semble encore une fois meilleur. En effet, avec des moyennes de temps de calcul d'au maximum 0.4s pour les deux premiers ensembles d'objectifs, cette configuration de l'algorithme donne de très bons résultats. Toutes les exécutions réalisées (60 par ensemble d'objectifs) retournent une bonne solution dans un temps relativement court et surtout très stable. L'écart type n'est que de 40ms et le minimum et maximum ne sont pas éloignés de plus de 150ms.

Pour la première fois, on arrive à trouver des solutions satisfaisant toutes les fonctions objectif pour l'ensemble d'objectifs S_{2agr} avec 10% d'exécutions donnant une bonne solution. L'ensemble d'objectifs S_{4agr} reste trop complexe. Mais cette fois-ci, pour les solutions retournées lorsque l'on atteint les 10 secondes, le nombre de fonctions objectif satisfaites dépasse la moitié et atteint en moyenne 60%. Pour un même ensemble d'objectifs le nombre de générations augmente de nouveau (pour un temps donné) puisque l'agrégation des objectifs permet de réduire le nombre de fonctions objectif. Et donc par la même occasion de réduire le coût d'une génération.

Synthèse pour l'exemple *Godaddy*

Godaddy est un exemple de plate-forme d'e-commerce. Par rapport à nos objectifs définis dans la section de mise en œuvre, 22 éléments ont été identifiés dans une de ses pages. Ces 22 éléments ont été associés à leur couleur afin de construire 22 variables (paires élément-caractéristique) sur lesquelles les algorithmes se sont basés pour trouver des solutions d'adaptation. NSGA-II et NSGA-III ont un schéma commun mais divergent aussi beaucoup sur certains points clé.

Sur ce site de taille moyenne, NSGA-II permet de trouver de bonnes solutions d'adaptation pour des ensembles d'objectifs pas trop complexes. Cependant, le temps de calcul, supérieur à 5 secondes, est assez important. Le fait de regrouper les objectifs en fonctions objectif (agrégation) permet de réduire le nombre de fonctions objectif et donc de réduire le coût d'une génération (itération) de l'algorithme. La contrepartie de l'agrégation pourrait être, entre autres, le risque de compensation d'un objectif par un autre (dans la même fonction objectif). Néanmoins, cet écueil a été évité dans notre implémentation (cf. 6.4.6). Cette agrégation apporte donc une amélioration notable en termes de temps de calcul. Les deux ensembles les plus complexes restent tout de même trop difficiles à traiter et on n'obtient aucune bonne solution en moins de 10 secondes.

Avec NSGA-III, qui est prévu pour manipuler davantage de fonctions objectif sans cette perte de performance caractéristique de la version antérieure, les résultats sont assez différents. Pour un même ensemble d'objectifs, NSGA-III permet de réduire le temps d'exécution et semble trouver une bonne solution en réalisant moins de générations. Le comportement général se rapproche de la version agrégée de NSGA-II sans pour autant faire appel à une quelconque agrégation des objectifs. Lorsque l'on agrège les objectifs avec NSGA-III, on en tire également des bénéfices. Les temps de calcul moyens sont très bas et surtout relativement stables. Il devient même possible de trouver quelques bonnes solutions pour un ensemble d'objectifs assez complexe qui jusque-là ne pouvait pas être résolu. Le dernier ensemble d'objectifs (le plus complexe) reste tout de même trop difficile, cependant la qualité des solutions retournées lorsque les 10 secondes sont atteintes est bien meilleure et dépasse les 60% de satisfaction.

Cette page d'exemple permet déjà d'avoir un aperçu des différentes versions et configurations des algorithmes NSGA-II et NSGA-III. Certaines hypothèses ont été émises comme par exemple une meilleure convergence de NSGA-III. D'autre exemples sont donc utiles afin de confirmer ou non ces hypothèses. La section suivante compare les résultats obtenus sur plusieurs autres pages Web de manière à étayer l'expérimentation et à tirer des conclusions globales.

Comparaison sur plusieurs pages Web

L'expérimentation sur la plate-forme d'e-commerce *Godaddy* a permis d'avoir un aperçu général sur le comportement des algorithmes et d'émettre certaines hypothèses. Cette section compare les résultats obtenus sur les pages provenant de plusieurs sites Web introduites précédemment. Les configurations des algorithmes et ensembles d'objectifs utilisés sont ceux définis dans la section de mise en œuvre et utilisés sur l'exemple de *Godaddy*.

Le tableau suivant donne les valeurs principales relatives aux différentes pages sur lesquelles vont être effectuées les comparaisons. Elles sont données par ordre croissant du nombre de variables.

TABLE 8.10 – Caractéristiques des pages Web étudiées

Site Web	Variables	S_1 Fcts. obj.	S_2 Fcts. obj.	S_3 Fcts. obj.	S_4 Fcts. obj.
Legibase	8	5	6	13	14
NFB	9	5	6	14	15
GDArchitect	10	8	9	18	19
BLMyAccount	18	9	10	26	27
Parempuyre	22	9	10	21	22
Godaddy	22	14	15	36	37
BLFamilyPortal	24	12	13	36	37
UBOFormation	24	17	18	41	42
UBOLEA	27	21	22	48	49
ReasonFrance	59	41	42	100	101

D'après les ensembles d'objectifs définis précédemment, certaines pages Web comme celle de Legibase ou celle de la NFB (National Federation of the Blind) n'ont que très peu de variables. Au contraire, une page comme celle du site ReasonFrance (très complexe) a un très grand nombre de variables et par conséquent beaucoup d'objectifs les connectant. Ce dernier exemple sera écarté de la comparaison puisque l'implémentation de NSGA-III ne permet pas d'obtenir des résultats du fait du trop grand nombre d'objectifs.. Une petite modification a été apportée pour traiter ce cas mais la comparaison directe ne peut plus être faite avec les autres exemples. Cet exemple sera donc détaillé un peu plus loin.

La première table (Table 8.11) compare l'utilisation de NSGA-II et NSGA-III pour les ensembles d'objectifs S_1 et S_3 sur les 9 pages Web. La page de *Godaddy* (étudiée en détails précédemment) est ajoutée dans ce tableau comparatif afin d'avoir un point de repère.

TABLE 8.11 – Comparaison globale (S_1 et S_3) sur plusieurs pages réelles (Temps/ exécutions $\geq$ 10s / satisfaction des exécution $\geq$ 10s)

Ensemble d'objectifs	S_1		S_3	
Algorithme	NSGA-II	NSGA-III	NSGA-II	NSGA-III
unités / Pages	s / % / %	s / % / %	s / % / %	s / % / %
Legibase	0.34 / 0 / –	0.12 / 0 / –	0.71 / 4 / 83.3	0.18 / 0 / –
NFB	0.39 / 0 / –	0.15 / 0 / –	0.46 / 0 / –	0.21 / 0 / –
GDArchitect	0.70 / 0 / –	0.25 / 0 / –	0.72 / 0 / –	0.30 / 0 / –
BLMyAccount	2.24 / 0 / –	0.94 / 0 / –	9.40 / 90 / 90	1.74 / 0 / –
Parempuyre	0.69 / 0 / –	0.29 / 0 / –	0.78 / 0 / –	0.41 / 0 / –
Godaddy	5.87 / 4 / 67.9	2.71 / 0 / –	9.89 / 94 / 69.3	3.53 / 0 / –
BLFamilyPortal	3.77 / 0 / –	2.53 / 0 / –	9.92 / 97 / 92.3	5.67 / 17 / 92.3
UBOFormation	8.97 / 54 / 61.8	5.88 / 0 / –	10 / 100 / 34.4	7.33 / 17 / 94.4
UBOLEA	10 / 100 / 32.4	9.95 / 87 / 86.7	10 / 100 / 16.4	10 / 100 / 75.5

Sur la page la plus simple de notre expérimentation (*Legibase*) le temps d'exécution moyen avec NSGA-II est relativement court pour S_1 comme pour S_3 (seulement quelques centaines de millisecondes). Ceci s'explique par le nombre de variables réduit et la faible

quantité d'objectifs définis. Néanmoins, l'utilisation de NSGA-III n'est pas sans intérêt. Avec NSGA-III, les temps de calcul sont grandement réduits pour passer en dessous de la barre des 0.2ms (S_1 et S_3). Pour l'ensemble d'objectifs S_3, le temps d'exécution est divisé par 4.

Si on s'intéresse à des pages de taille moyenne, on constate également une amélioration des temps de calcul avec NSGA-III. Sur la page *BLFamilyPortal* on passe d'une moyenne de 3.77s pour NSGA-II à 2.53s pour NSGA-III pour l'ensemble d'objectifs S_1. Pour l'ensemble d'objectifs S_3, on passe de 9.92s à 5.67s. Une différence importante se trouve dans le nombre d'exécutions qui donnent une bonne solution en moins de dix secondes. En effet, pour ce même exemple avec S_3, on passe de seulement 3% d'exécutions terminant en moins de 10s pour NSGA-II à 83% avec NSGA-III.

Avec la page Web UBOLEA, aucune bonne solution n'est trouvée avec NSGA-II, ni pour S_1, ni pour S_3. NSGA-III permet de trouver des bonnes solutions dans 13% des cas pour l'ensemble d'objectifs S_1. Mais surtout, que ce soit pour un ensemble d'objectifs ou un autre, NSGA-III permet d'augmenter considérablement le niveau de satisfaction des solutions retournées lorsque les dix secondes sont atteintes. Ceci ne se vérifie pas uniquement sur cette page Web mais sur plusieurs exemples.

Le tableau suivant (cf. Table 8.12) contient les résultats obtenus avec les ensembles d'objectifs S_2 et S_4. Ces deux ensembles d'objectifs sont les deux ensembles les plus complexes de notre expérimentation.

TABLE 8.12 – Comparaison globale (S_2 et S_4) sur plusieurs pages réelles (Temps/ exécutions $\geq$ 10s / satisfaction des exécutions $\geq$ 10s

Ensemble d'objectifs	S_2		S_4	
Algorithme	NSGA-II	NSGA-III	NSGA-II	NSGA-III
unités / Pages	s / % / %	s / % / %	s / % / %	s / % / %
Legibase	10 / 100 / 48.5	5.80 / 20 / 90.8	10 / 100 / 40.7	7.16 / 44 / 87.1
NFB	10 / 100 / 58.6	3.81 / 4 / 92.9	10 / 100 / 56	6.49 / 34 / 92.7
GDArchitect	10 / 100 / 35	9.20 / 80 / 88.3	10 / 100 / 34.7	9.47 / 87 / 85.8
BLMyAccount	10 / 100 / 22.3	10 / 100 / 61.9	10 / 100 / 23.3	10 / 100 / 58.5
Parempuyre	10 / 100 / 31.4	9.95 / 97 / 82.9	10 / 100 / 33.2	10 / 100 / 79.5
Godaddy	10 / 100 / 20	10 / 100 / 43.1	10 / 100 / 21.4	10 / 100 / 41.4
BLFamilyPortal	10 / 100 / 21.7	10 / 100 / 43.1	10 / 100 / 21.6	10 / 100 / 41.9
UBOFormation	10 / 100 / 19.3	10 / 100 / 37.8	10 / 100 / 19.5	10 / 100 / 37.9
UBOLEA	10 / 100 / 17.7	10 / 100 / 34.8	10 / 100 / 18.2	10 / 100 / 34.1

Ici, aucune des configurations (site Web - ensemble d'objectifs) ne donne lieu à une bonne solution d'adaptation avec NSGA-II. Les ensembles d'objectifs contiennent un grand nombre d'objectifs (cf. Table 8.10) et sans agrégation on a autant de fonctions objectif que ce qu'il y a d'objectifs. NSGA-III permet, pour l'ensemble d'objectifs S_2, de trouver quelques bonnes solutions en moins de dix secondes pour certaines pages Web (les plus petites et moins complexes). Par contre, ce tableau montre de manière évidente que NSGA-III permet d'augmenter de manière significative le niveau de satisfaction des solutions retournées une fois les dix secondes atteintes. Parfois même on atteint des valeurs proches

de 100% (S_2 avec NFB) ce qui signifie que les solutions retournées ne sont pas si mauvaises que cela et qu'il ne manque peut-être pas grand chose pour en trouver qui soient pleinement satisfaisantes.

TABLE 8.13 – Comparaison globale (S_{1agr} et S_{3agr}) sur plusieurs pages réelles (Temps/ exécutions $\geq$ 10s / satisfaction des exécutions $\geq$ 10s)

Ensemble d'objectifs	S_{1agr}		S_{3agr}	
Algorithme	NSGA-II	NSGA-III	NSGA-II	NSGA-III
unités / Pages	s / % / %	s / % / %	s / % / %	s / % / %
Legibase	0.27 / 0 / –	0.10 / 0 / –	0.21 / 0 / –	0.13 / 0 / –
NFB	0.22 / 0 / –	0.13 / 0 / –	0.24 / 0 / –	0.17 / 0 / –
GDArchitect	0.22 / 0 / –	0.15 / 0 / –	0.31 / 0 / –	0.17 / 0 / –
BLMyAccount	0.52 / 0 / –	0.32 / 0 / –	0.90 / 4 / 66.7	0.38 / 0 / –
Parempuyre	0.24 / 0 / –	0.18 / 0 / –	0.24 / 0 / –	0.19 / 0 / –
Godaddy	0.41 / 0 / –	0.38 / 0 / –	0.46 / 0 / –	0.40 / 0 / –
BLFamilyPortal	0.48 / 0 / –	0.44 / 0 / –	0.83 / 4 / 66.7	0.52 / 0 / –
UBOFormation	0.47 / 0 / –	0.45 / 0 / –	1.46 / 10 / 66.7	0.45 / 0 / –
UBOLEA	2.78 / 24 / 45	1.82 / 14 / 50	10 / 100 / 20	10 / 100 / 60

L'agrégation avait fait ses preuves sur la page de *Godaddy*. Elle a été mise en place sur les autres pages Web (tables 8.13 et 8.14). Sur certaines pages avec l'ensemble d'objectifs S_{1agr} l'amélioration apportée par NSGA-III est assez limitée (environ 10%). Sur certaines autres, elle est bien plus marquée (50% voire plus). Avec UBOLEA et l'ensemble d'objectifs le plus complet (S_{4agr}), aucune solution n'est trouvée que ce soit avec l'une ou l'autre des deux versions de NSGA. Cependant NSGA-III permet de tripler le niveau de satisfaction des solutions retournées lorsque l'on atteint le temps limite d'exécution, passant de 20% à 60%. De la même manière sur UBOFormation (avec S_{3agr}) où le niveau de satisfaction sur les 10 exécutions non terminées était d'environ 67%, avec NSGA-II, avec NSGA-III toutes les exécutions retournent des solutions satisfaisant toutes les fonctions objectif.

TABLE 8.14 – Comparaison globale (S_{2agr} et S_{4agr}) sur plusieurs pages réelles (Temps/ exécutions $\geq$ 10s / satisfaction des exécutions $\geq$ 10s

Ensembles d'objectifs	S_{2agr}		S_{4agr}	
Algorithme	NSGA-II	NSGA-III	NSGA-II	NSGA-III
unités / Pages	s / % / %	s / % / %	s / % / %	s / % / %
Legibase	7.17 / 70 / 52.5	0.96 / 4 / 75	8.13 / 80 / 54	1.89 / 4 / 80
NFB	4.25 / 37 / 55	0.58 / 0 / –	7.23 / 70 / 54	2.41 / 10 / 80
GDArchitect	6.87 / 67 / 47.5	1.36 / 7 / 75	6.59 / 64 / 64	3.00 / 20 / 80
BLMyAccount	8.50 / 84 / 47.5	4.83 / 44 / 72.5	10 / 100 / 34	9.72 / 97 / 72
Parempuyre	7.52 / 74 / 47.5	1.97 / 14 / 75	8.16 / 80 / 60	4.34 / 37 / 80
Godaddy	10 / 100 / 15	9.11 / 90 / 62.5	10 / 100 / 26	10 / 100 / 60
BLFamilyPortal	10 / 100 / 30	8.19 / 80 / 65	10 / 100 / 22	10 / 100 / 60
UBOFormation	10 / 100 / 22.5	9.40 / 94 / 60	10 / 100 / 40	9.41 / 94 / 68
UBOLEA	10 / 100 / 20	9.70 / 97 / 55	10 / 100 / 26	10 / 100 / 46

Les deux ensembles d'objectifs S_{2agr} et S_{4agr} sont ceux pour lesquels trouver une bonne solution est le plus difficile. La table 8.14 montre l'effet de l'agrégation sur les 9 pages Web. Alors que NSGA-II peine à donner de bonnes solutions dans la plupart des cas, NSGA-III affiche des temps de calcul plus faibles. Cette troisième version permet aussi de trouver de bonnes solutions là où cela n'était pas le cas avec la version précédente.

Synthèse générale sur NSGA-II et NSGA-III

Les tendances se dessinent et les hypothèses émises avec l'exemple *Godaddy* semblent se confirmer. L'algorithme d'optimisation NSGA-II permet de trouver des solutions (adaptations) qui satisfont tous les objectifs utilisateur. Cependant des limitations inhérentes à son fonctionnement propre ne permettent pas de traiter des pages Web de grande taille (contenant beaucoup d'éléments). De plus, si le nombre d'objectifs définis par l'utilisateur devient important, cela engendre également une augmentation des dépendances entre objectifs et par conséquent, une difficulté accrue. Dans cette expérimentation, l'ensemble des exemples est traitable, c'est-à-dire qu'une bonne solution d'adaptation existe. Pour un ensemble d'objectifs donné, lorsqu'aucune tentative ne donne de bonne solution, cela signifie simplement qu'elle n'a pas pu en trouver une (même si il en existe au moins une). Dans la réalité, la solution idéale (qui satisfait tout les objectifs) n'existe pas forcément et la solution la plus satisfaisante devra être conservée.

NSGA-III offre une bonne amélioration dans l'ensemble. Cette version de l'algorithme qui utilise des opérateurs génétiques un peu différents permet de converger vers une bonne solution en un nombre de générations (itérations) bien plus petit. Cela signifie qu'avec un même ensemble d'objectifs, une bonne solution est trouvée en réduisant le nombre d'opérations réalisées sur les individus de la population (ensemble de solutions). Parmi les modifications apportées par NSGA-III, l'amélioration de la diversité des individus lors de la sélection peut justifier d'une meilleure convergence de l'algorithme en permettant de meilleurs croisements. Au final, le niveau de satisfaction général des solutions est supérieur à ce que l'on obtient avec NSGA-II. Un rapport de 2 est souvent constaté sur le temps d'exécution ou bien sur le niveau de satisfaction.

Les figures 8.1 (a) à (d) mettent en évidence ce phénomène respectivement sur les ensembles d'objectifs S_4 et S_{4agr} en ce qui concerne les pourcentages d'objectifs satisfaits et sur les ensembles d'objectifs S_1 et S_{1agr} en ce qui concerne les temps d'exécution. Dans ces graphiques, l'abscisse représente le nombre d'objectifs associés aux différentes pages Web de notre corpus. Dans les graphiques (a) et (b) (haut), l'ordonnée représente le pourcentage d'objectifs satisfaits pour NSGA-III (le plus élevé) divisé par le pourcentage d'objectifs satisfaits pour NSGA-II. Dans les graphiques (c) et (d) (bas), l'ordonnée représente le temps moyen d'exécution pour NSGA-II (le plus élevé) divisé par le temps moyen d'exécution pour NSGA-III.

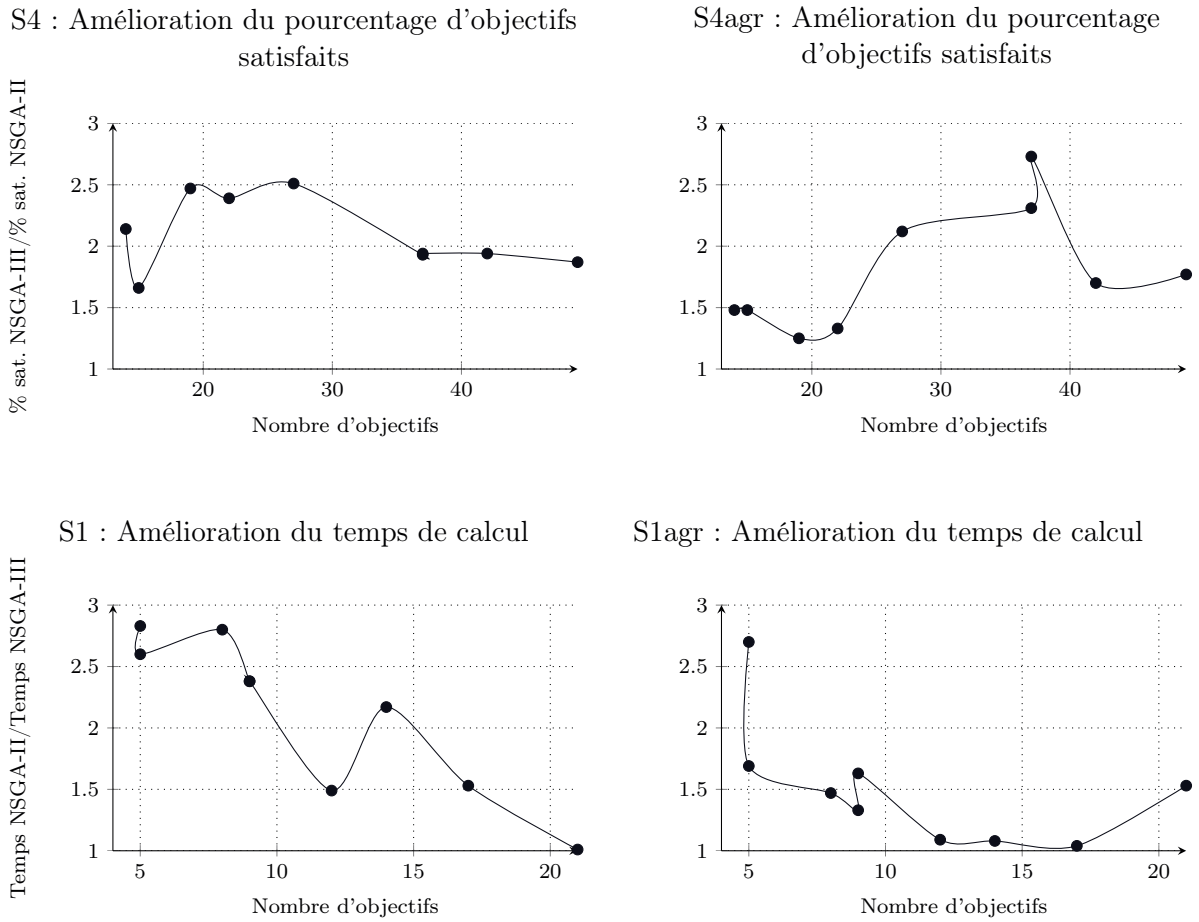


FIGURE 8.1 – Amélioration apportée par NSGA-III sur les pourcentages d'objectifs satisfaits et le temps de calcul

Remarque : Certains points semblent avoir la même abscisse et une ordonnée différente. C'est en fait le cas. Sur le dernier graphique, nous avons par exemple deux points à 9 objectifs. Dans notre corpus de pages nous avons deux pages qui ont neuf objectifs pour l'ensemble d'objectifs S_1, ces deux pages ne donnant pas exactement les mêmes résultats en termes de temps de calcul.

L'agrégation permet de traiter la limitation bien connue de NSGA-II, qui perd en performance au fur et à mesure que le nombre de fonctions objectif augmente. Elle permet de réduire le nombre de fonctions objectif et par conséquent de réduire le coût d'une génération. Cependant on observe pour NSGA-III, qui est en soi prévu pour manipuler beaucoup de fonctions objectif, que l'agrégation apporte également une amélioration. La réduction du nombre de fonctions objectif est donc positive dans les deux algorithmes.

NSGA-III - Utiliser beaucoup de fonctions objectif

Dans l'article [JD14a] sur lequel nous nous sommes basés pour implémenter NSGA-III, ainsi que l'implémentation non officielle réalisée [Chi], la taille de la population dépend de plusieurs paramètres. À la différence de NSGA-II, elle n'est pas passée en paramètre et connue à l'avance. Cette dernière dépend en effet du nombre de divisions que l'on choisit

d'avoir pour chaque fonction objectif et donc aussi du nombre lui-même de fonctions objectif. Pour simplifier, la taille de la population est proportionnelle (à coefficient constant) au nombre de points de référence.

Avec la page *ReasonFrance*, que nous n'avons pas fait apparaître dans les tableaux précédents, le nombre de fonctions objectif peut atteindre la centaine (sans agrégation). La figure 6.2 montre la construction des points de références. Dans le cas de cette page, le nombre de points de référence serait de 5151 d'après la formule suivante :

$$Rps = \binom{M+p-1}{p} \tag{8.6}$$

D'après les documents qui ont servi pour l'implémentation, la population doit être supérieure au nombre de points de référence et de plus elle doit être divisible par 4. Si on applique ces règles, la plus petite taille de population supérieure ou égale au nombre de points de référence et divisible par 4 est 5152. Le problème est que, que ce soit avec NSGA-II ou NSGA-III, la complexité algorithmique est de MN^2 avec N la taille de la population. Avec une population de plus de 5000 individus, une fois élevée au carré, on doit réaliser une grande quantité d'opérations. D'une part les dix secondes ne sont pas suffisantes pour effectuer ne serait-ce qu'une seule génération, mais surtout le temps qu'il faudrait si jamais on voulait tout de même la calculer entièrement serait trop long. Au bout d'une heure d'exécution sur la plate-forme utilisée pour les expérimentations, elle n'était toujours pas calculée. Sachant qu'il faut plusieurs générations pour converger vers une bonne solution, il n'est même pas envisageable de tenter ce calcul même s'il s'agit d'un processus d'arrière-plan.

Une des solutions proposée est de limiter la taille maximale de la population en dépit des recommandations, puisque le nombre de divisions ne peut pas être réduit davantage.

8.3.4 Impact du choix des paramètres

Variation de la taille de population (NSGA-II)

La taille de la population, c'est-à-dire le nombre de solutions traitées à chaque génération, a une grande importance sur les performances de l'algorithme. Elle intervient de manière significative dans la complexité de l'algorithme. En effet, cette composante est élevée au carré (MN^2) quand M est le nombre de fonctions objectifs et N la taille de la population. Il est donc crucial de choisir une taille de population de manière à avoir d'une part le plus d'interactions diverses entre individus et d'autre part un nombre suffisant de générations.

Pour augmenter le nombre et la diversité des opérations de croisement et de sélection, la population doit être d'une certaine taille. Plus la taille de population sera importante et plus la diversité des croisements le sera. La contre-partie majeure à l'agrandissement de la population, est la réduction du nombre de générations (pour un temps donné). Dans ce contexte, étant donné la complexité de l'algorithme pour une génération, multiplier par deux la taille de la population revient à diviser par quatre le nombre de générations

(ordre de grandeur). Le nombre de générations correspondant au nombre de descendances générées, plus ce dernier sera important et plus on aura de chances de converger vers une bonne solution. Il est donc crucial de trouver le bon compromis, avoir la taille de population la plus grande possible tout en conservant un nombre de générations suffisant.

La taille de la population est un paramètre constant et fixé au début de l'exécution. Les courbes de la figure 8.2 montrent ce qui se passe lorsque l'on exécute NSGA-II avec des tailles de populations diverses.

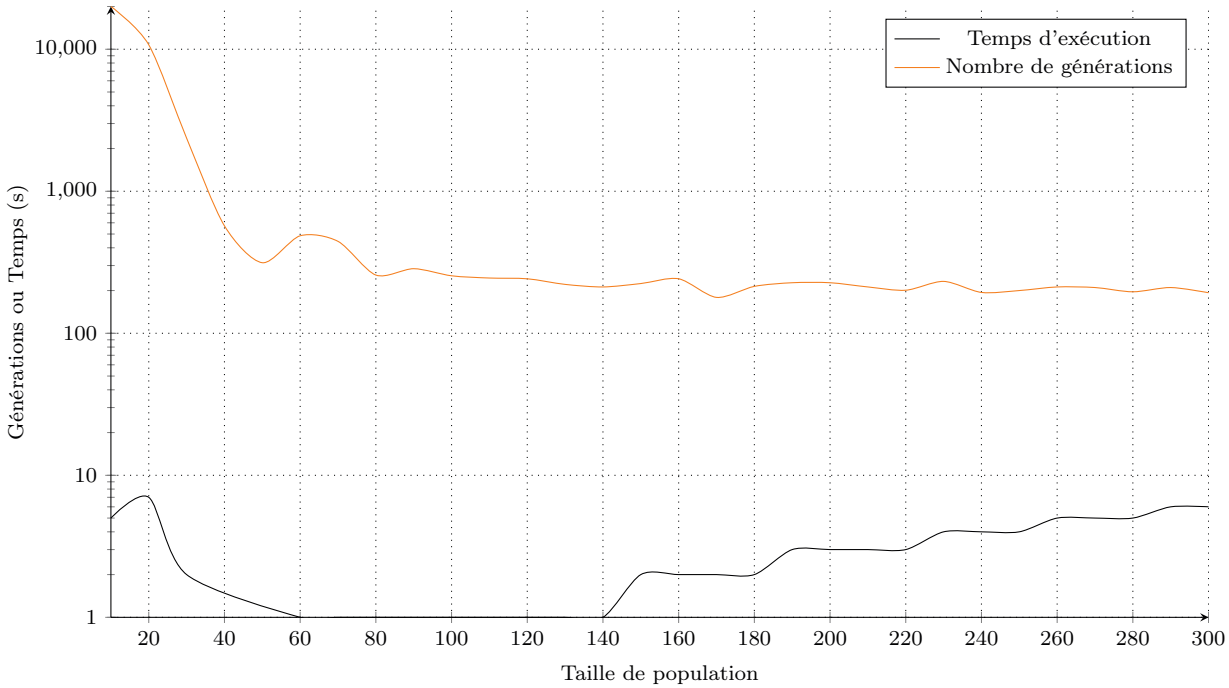


FIGURE 8.2 – Temps d'exécution / Génération en fonction de la taille de population (Exemple : Godaddy S_1)

On peut observer que le temps d'exécution (pour atteindre une bonne solution) et le nombre de générations sont élevés quand la population est de trop petite taille (20). Au-delà d'une taille de population de 250 individus, le temps d'exécution augmente sans exagérer et le nombre de générations reste relativement stable. Le temps d'exécution est très faible pour les tailles de population de 60 à 140. Cependant, une étude similaire sur d'autres sites nous a conduit dans l'article [BHM14a] à choisir la taille de 250 individus par génération.

Impact du taux de croisement et de mutation

Le taux de croisement et de mutation fait aussi partie des paramètres importants d'un algorithme évolutionnaire. Il détermine la quantité d'individus qui vont potentiellement être affectés par une mutation, c'est une part de l'aléatoire que l'on autorise dans l'exécution de l'algorithme. Plus le taux est élevé et plus cette part d'aléatoire est petite. Par exemple si on a un taux de 0.94 (ou 94%), alors uniquement 6% (100% - 94%) des

croisements seront suivis d'une mutation.

Si le taux est fixé à une valeur proche de 0 alors on se trouve dans le cas d'un algorithme basé fortement sur de l'aléatoire. Sauf que dans ce cas on s'autoriserait à tester plusieurs fois la même solution, puisqu'une fois testée, la solution n'est pas exclue de l'espace de recherche. Si le taux atteint 1, alors aucune mutation n'aura lieu, seuls les croisements permettront de faire évoluer la population pour atteindre de nouvelles solutions. Cette configuration peut être viable pour certains problèmes, mais peut aussi être à l'origine de stagnation autour d'un maximum local, l'opérateur de mutation permettant de sélectionner des solutions au hasard et potentiellement d'atteindre des nouvelles zones encore non explorées de l'espace de recherche.

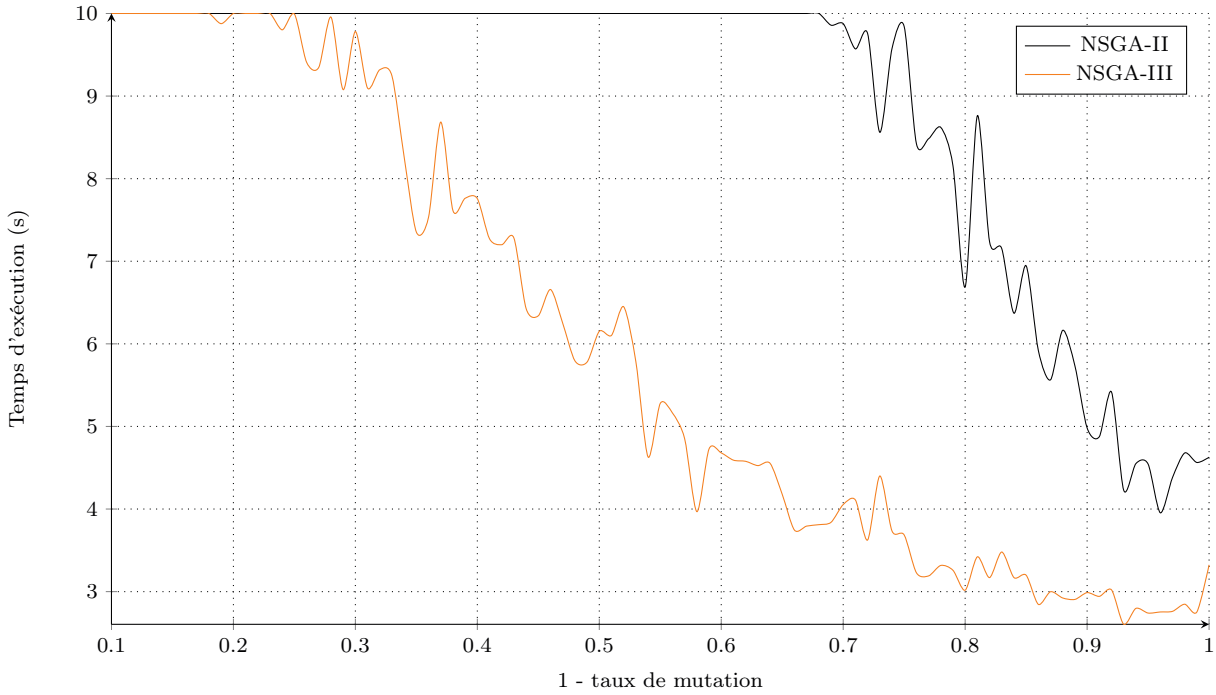


FIGURE 8.3 – Temps d'exécution en fonction de (1 - taux de mutation) (Godaddy S_1)

Le graphique 8.3 montre l'impact de différentes valeurs du taux de croisement sur le temps nécessaire pour trouver au moins une bonne solution. Deux choses principales sont à remarquer, la première concerne la relation entre le temps d'exécution et la valeur du taux choisie, la seconde concerne l'algorithme utilisé. Les deux courbes suivent la même tendance, lorsque la part d'aléatoire diminue (le taux de croisement augmente), le temps mis en moyenne pour trouver au moins une bonne solution diminue aussi. Sur l'exemple Godaddy où seul l'objectif de contraste entre le texte et l'arrière-plan est pris en compte, on peut constater un léger rebond lorsque l'on s'approche de la valeur 1 (pas de mutation). Selon le problème étudié, l'approche de la valeur 1 peut se matérialiser comme une stabilisation ou un rebond (voire même une amélioration dans certains cas). Sur la courbe correspondant à NSGA-III, la stabilisation puis la remontée en 1 est assez bien marquée. On observe une différence nette entre les deux courbes, celle qui correspond à NSGA-III entame sa descente plus rapidement que celle qui correspond à NSGA-II. De plus pour toute valeur supérieure à environ 0.3, le temps moyen pour NSGA-III est plus faible que

celui de NSGA-II. Une étude sur différentes configurations a ainsi permis de mettre en évidence qu'une valeur moyenne pour ce taux de croisement/mutation de 0,94 permettait d'obtenir le temps le plus faible tout en donnant de bons résultats.

Choix du nombre de points de références

NSGA-III est basé sur la répartition d'un certain nombre de points de référence dans l'espace. L'objectif premier de l'utilisation de ces points de référence est d'obtenir une meilleure diversité des individus choisis lors de la sélection des parents. Cette diversité accrue (par rapport à NSGA-II) est obtenue en prenant en compte un plus grand nombre d'individus dans l'espace mais aussi en ayant une meilleure précision.

Plus le nombre de points de référence utilisés est important et plus la précision (localisation des individus dans l'espace) sera grande. Cependant, telle qu'est définie et conçue cette version de l'algorithme, le nombre de points de référence conditionne une taille de population. Pour que l'exécution soit efficace, le nombre d'individus dans la population doit être supérieur au nombre de points de référence. De plus la quantité de points est calculée en fonction du nombre de fonctions objectif et du nombre de divisions que l'on souhaite avoir pour chacune d'entre elles. Lorsque le nombre de fonctions objectif devient important, même si le nombre de divisions reste faible, alors la quantité de points de référence devient elle aussi très importante. Une grande population a pour effet de fortement réduire l'efficacité de l'algorithme en raison de sa complexité.

Le nombre de fonctions objectif doit par conséquent être limité afin de conserver un nombre de points de référence, et donc d'individus raisonnable dans la population. Le terme de raisonnable indique ici un nombre d'individus suffisamment bas pour ne pas casser l'exécution de l'algorithme du fait du facteur « carré » (^{2}) qui porte sur la taille de la population (cf. complexité algorithmique de NSGA-III). Il n'existe donc pas de valeur précise à donner, cela dépend notamment de la puissance de calcul dont on dispose, mais aussi du problème que l'on est en train de traiter.

8.4 Conclusion

Un corpus de divers types de pages Web réelles a été utilisé afin de réaliser des expérimentations. Pour chacune de ces pages, plusieurs types d'objectifs (souhaits de l'utilisateur) ont été utilisés et combinés. En plus de ces combinaisons d'objectifs pour chaque page, nous avons joué sur les paramètres des algorithmes évolutionnaires afin d'analyser leur impact et les bénéfices/pertes en fonction des pages. Ces expérimentations ont été menées à l'aide du démonstrateur développé. Alors que l'approche exacte ne permet pas de résoudre de tels problèmes en un temps raisonnable, de bons résultats ont été obtenus sur des petites pages mais aussi sur des pages de taille moyenne à l'aide d'algorithmes évolutionnaires. La comparaison entre NSGA-II et NSGA-III montre que la dernière version de NSGA apporte une amélioration notable sur les temps de calculs lorsqu'il existe une solution satisfaisant tous les objectifs mais aussi sur le niveau de satisfaction des solutions incorrectes. L'agrégation de fonctions objectif est aussi un facteur d'amélioration important.

Chapitre 9

Conclusion et perspectives

9.1 Rappel du contexte

Le Web est de nos jours devenu incontournable. La quantité de pages Web disponibles ne cesse d'augmenter. De plus en plus de services naissent, nous permettant de réaliser plus facilement des opérations de la vie quotidienne. Malheureusement la démocratisation de cet outil entraîne de plus en plus la suppression des services « physiques » qui deviennent maintenant totalement virtuels. L'accessibilité de ces services (au sens large du terme) par le plus grand nombre est donc un enjeu non négligeable. Le désir des entreprises de déployer leurs propres services augmentant, des aberrations sont parfois visibles. Par exemple, certaines plates-formes de messagerie permettant de créer une adresse e-mail demandent une autre adresse e-mail de secours. Ce champ étant parfois obligatoire, cela signifie que l'on ne peut pas créer sa première adresse e-mail sur ces sites.

L'accessibilité des pages Web et des divers services pour les personnes déficientes visuelles n'est, de nos jours, pas encore réellement considérée et fait souvent l'objet d'ignorance voire même de mépris. Pourtant lorsqu'une personne perd la vue, beaucoup de portes se ferment devant elle, et le monde du Web et des nouvelles technologies participent grandement à l'exclusion des personnes déficientes visuelles. Il existe tout de même des outils permettant d'améliorer la qualité de l'accès à l'information. Ces outils sont également appuyés par des standards et recommandations qui donnent des règles de conduite pour le développement de pages et services web. Des outils de lecture d'écran permettent par exemple aux personnes non-voyantes d'accéder de manière vocale (ou même en braille) à l'information textuelle. Même si les personnes qui ont une basse vision sont tout à fait en capacité d'utiliser les lecteurs d'écran, le reste de vue dont elles disposent reste tout de même important pour elles. Une vision, y compris fortement dégradée, tant qu'elle peut permettre l'accès à de l'information, reste le principal moyen d'accès. Des outils existent donc pour transformer, adapter, améliorer l'apparence des pages Web. Les agrandisseurs d'écran (aussi appelés *magnifiers* d'écran) permettent entre autres d'appliquer des filtres de couleurs sur l'écran.

Ces agrandisseurs d'écran apportent de vraies améliorations sur le confort d'utilisation de l'outil informatique de manière générale. Cependant, lorsqu'il s'agit de les utiliser sur des pages Web, les bénéfices sont plus mitigés. En effet, la diversité des styles d'affichage

de pages Web, notamment due à la liberté que les langages HTML et CSS offrent aux développeurs, rend bien plus complexe l'adaptation des pages. Les filtres de couleurs que fournissent les agrandisseurs d'écran permettent très bien de résoudre des problèmes tels que l'éblouissement lorsque celui-ci se trouve sur l'intégralité de l'écran. Mais lorsque l'on est sur une page Web, alors on peut facilement avoir des parties différentes avec des styles complètement opposés. Les filtres de couleurs s'appliquant sur l'intégralité de l'écran, il est très difficile voire même impossible d'adapter une page dans son ensemble.

Des travaux de recherche plus spécifiques au Web tentent de combler les manques et de dépasser les limites des outils d'assistance existants. Des recherches ont été réalisées autour du daltonisme et sur l'adaptation des pages en fonction du type de daltonisme de la personne. Des filtres de couleurs sont appliqués sur la page pour transformer certaines couleurs existantes (celles qui posent problème) et les remplacer par d'autres couleurs. Ces filtres de couleurs appliqués en fonction du type de daltonisme de la personne sont définis en mémoire et simplement appliqués à la page. Dans le cas de la malvoyance en général, il n'est pas possible de définir des filtres de couleurs généraux pour chaque type de pathologie. De plus chez chaque individu, une pathologie a une évolution qui lui est propre, d'autre part, plusieurs pathologies peuvent être combinées. D'autres travaux de recherche explorent une alternative en offrant la possibilité à l'utilisateur d'indiquer ses besoins indépendamment de sa pathologie. Ces outils permettent ensuite d'appliquer des modifications en fonction des besoins de l'utilisateur. Ces travaux de recherche sont limités sur certains points, par exemple les transformations pouvant être appliquées sur les pages restent limitées à des éléments très simples comme le texte, les liens ou les images.

Les travaux de recherche que nous avons menés ont permis de proposer une nouvelle approche qui diffère des travaux existants. Les besoins réels et personnels de l'utilisateur sont placés au cœur de nos recherches. Ceci, afin de proposer des adaptations les plus cohérentes possibles avec les besoins propres d'un utilisateur qui de plus peuvent évoluer dans le temps. Des résultats prometteurs ont également été obtenus sur des pages de petite taille et même sur un certain nombre de pages de taille moyenne. Les méthodes de recherche d'une bonne adaptation permettent de réaliser des compromis entre les besoins parfois contradictoires d'une personne mais aussi avec l'apparence originale de la page. Le fait de prendre en compte l'apparence originale de la page permet entre autres de conserver le contexte (notamment de couleur) propre à une marque et choisie par le concepteur. Cela permet également de conserver au maximum la signature graphique originelle du site Web.

9.2 Contributions

Différentes étapes sont nécessaires dans l'élaboration d'un processus d'adaptation complet. Plusieurs domaines ont été abordés. Ces travaux de recherche, même encore incomplets, contribuent sur plusieurs points à la recherche scientifique.

Des notions importantes sur la vision humaine, la perception des couleurs et les différentes propriétés des couleurs sont détaillées. Nous avons également créé des représentations schématiques montrant l'impact de diverses pathologies affectant la vision. Ces schémas permettent d'observer et de se rendre compte de l'impact d'une pathologie sur la perception du monde extérieur et donc aussi sur l'utilisation de l'outil informatique. En-

suite, une étude a été réalisée autour des technologies d'assistance existantes et autour des différents travaux de recherche dans le domaine. Cela a permis de déterminer les manques et les limites de ces approches. Une analyse des besoins a été menée auprès de quelques personnes malvoyantes. L'objectif était de mettre en évidence un certain nombre de besoins des personnes qui ont une basse vision. Cette étude a effectivement permis de mettre en avant des problèmes et des besoins dont certains semblent récurrents. Un langage de représentation des préférences de l'utilisateur a été défini dans le but de formaliser ces besoins. Ces préférences sont ensuite utilisées dans des algorithmes de résolution pour tenter de trouver une bonne adaptation de la page. Une étude a été menée autour des différents algorithmes de résolution pouvant être utilisés. Plusieurs d'entre eux ont été retenus et utilisés. Des algorithmes comme ceux issus de la théorie des préférences ou des algorithmes évolutionnaires (des variantes de NSGA) ont donc été mis en œuvre.

Pour obtenir un processus d'adaptation complet (de la page d'origine jusqu'à la page modifiée) nous avons modélisé le problème à résoudre sur un corpus de pages Web réelles. Un démonstrateur a été développé de manière à automatiser la recherche d'une adaptation. Ce démonstrateur intégre deux algorithmes génétiques (algorithmes évolutionnaires) et propose de nombreux paramétrages. Il a ensuite été utilisé afin de réaliser des expérimentations approfondies sur plusieurs pages Web réelles. L'expérimentation permet entre autres de comparer la résolution par le biais d'algorithmes évolutionnaires avec ce que l'on pourrait obtenir à l'aide d'une approche exacte. De plus l'implémentation de deux versions d'algorithmes génétiques a permis de réaliser des comparaisons entre les deux et d'étudier les différences de comportement. Nous avons atteint un premier objectif d'adaptation pour des pages de taille moyenne dans un temps raisonnable.

9.3 Perspectives ouvertes

De nombreuses perspectives d'évolution s'ouvrent à la suite de ces travaux de recherche, notamment en rapport avec l'automatisation complète du processus de transformation d'une page, des méthodes de résolution utilisées et d'une expérimentation tournée vers les utilisateurs.

9.3.1 Acquisition des préférences

Jusqu'ici certaines étapes sont réalisées manuellement comme :

— en amont du processus la récupération des données de la page Web d'origine utiles pour le processus, l'acquisition des préférences de l'utilisateur et leur instanciation sur les données de la page ;

— en aval de la chaîne de transformation, la modification de la page une fois qu'une adaptation a été calculée.

L'acquisition des préférences n'est pas quelque chose d'évident, il n'est pas envisageable de demander à quelqu'un d'expliciter l'ensemble de ses préférences et de donner des valeurs précises (taille minimale ou maximale, niveau de contraste, etc.). Une telle demande serait décourageante et détournerait l'utilisateur de l'outil. Il n'existe pas non plus

de schéma général des déficiences sur lequel s'appuyer, même si pour certains cas précis, comme le daltonisme, des schémas d'adaptation standards ont pu être définis [San09]. Une autre difficulté vient du décalage entre les objets codés en HTML et les objets perçus par l'utilisateur. Là où l'utilisateur perçoit un menu et peut souhaiter appliquer au concept de menu une certaine adaptation, en HTML, on trouvera un codage sous forme de DIV (DIVision), de texte brut ou de marques de listes (LI, UL). De même des objets proches sur la page peuvent être éloignés dans le codage. Des méthodes permettant de recueillir de manière la plus transparente possible ces préférences, du point de vue de la perception de l'utilisateur et non de la conception technique de la page doivent donc être mises en œuvre. Nous envisageons de travailler à l'aide de tests visuels et d'apprentissage à partir d'exemples (par présentation de pages Web). Nous pensons également chercher dans les connaissances médicales des informations sur les relations possibles / impossibles entre certaines préférences afin de faciliter et de guider leur expression.

La récupération des éléments dans la page d'origine (choix des éléments à prendre ou à laisser) dépend des préférences de l'utilisateur. Notre approche se basant sur des préférences portant sur des éléments de haut niveau (menu, en-tête, contenu, etc.), il devient vite complexe d'identifier ces éléments sur les pages selon le langage de développement utilisé comme indiqué plus haut. Une méthode d'extraction automatique est donc nécessaire pour automatiser le processus complet. Cette méthode d'extraction automatique devra faire suite à l'identification des structures logiques de la page (lorsque le langage utilisé ne les définit pas explicitement). Un premier travail dans cette direction a été effectué par Franck Petitdemange dans le cadre d'un mémoire de Master Recherche [Pet14] et sera poursuivi.

9.3.2 Méthodes de résolution

Nous avons principalement exploré une approche par méta-heuristiques multi-objectifs. Nous envisageons plusieurs angles d'attaque pour améliorer cette première approche :

— Un premier axe consisterait à travailler sur la détection des cas difficiles pour avoir des méthodes alternatives. Par exemple, nous avons pu noter que la recherche de bonnes couleurs est coûteuse en temps et nous pourrions privilégier des solutions qui ont des bons résultats pour les fonctions objectif liées à la couleur dans les croisements et la sélection,

— Un second axe consisterait à combiner des résultats partiels de plusieurs exécutions de méta-heuristiques,

— Un troisième axe se préoccuperait d'adopter des stratégies d'adaptation différentes, notamment en recherchant des adaptations sur des parties de pages, puis en les ré-assemblant avant de finaliser l'adaptation sur les parties frontières,

— Enfin un quatrième axe se concentrerait sur l'apprentissage à partir d'adaptations déjà réalisées et mémorisées.

De plus, nous aimerions explorer d'autres méthodes de résolution. Le problème d'adaptation d'une page se prête bien à une modélisation sous la forme d'un problème de satisfaction de contraintes (CSP). Pour obtenir des résultats intéressants, la représentation de certaines propriétés comme celles des couleurs devra certainement faire l'objet d'une

transformation. Par exemple, pour la brillance d'une couleur, si on représente une couleur par ses trois composantes de base, alors si on fait varier les trois composantes dans l'ordre (on en augmente une jusqu'à la valeur maximum avant d'augmenter l'autre), la brillance n'évoluera pas de manière monotone. Ceci est dû au fait que les trois composantes de base n'ont pas la même brillance perçue. Une perte de performance importante dans les solveurs de contraintes peut être due à ce comportement.

Une autre méthode comme le Deep Learning ou une méthode similaire pourrait être utilisée pour résoudre ce même problème d'adaptation de page. Le fait de prendre le problème à des niveaux différents et de combiner les résultats pourrait être extrêmement bénéfique. Le Deep learning donne de bons résultats dans pas mal de domaines notamment la reconnaissance d'images [Pib15].

9.3.3 Validation par les utilisateurs concernés

Au-delà de la validation quantitative que nous avons effectuée dans nos expérimentations, nous aimerions effectuer une validation qualitative, intégrant davantage l'utilisateur final dans le processus de validation. Nous souhaitons en effet mettre en place de véritables tests utilisateur afin de recueillir des retours au fil de l'évolution des travaux. Cela nécessite de définir des indicateurs de qualité pertinents (pour juger des résultats), d'identifier des personnes qui effectueront les tests, de consolider notre corpus de pages à tester (en liaison avec les personnes effectuant les tests) et de définir des protocoles de tests.

Bibliographie

[AMS14] Sébastien Aupetit, Nicolas Monmarché, and Mohamed Slimane. Comparison of two swarm intelligence optimization algorithms on the textual color problem for web accessibility. In *Swarm Intelligence Based Optimization - First International Conference, ICSIBO 2014, Mulhouse, France, May 13-14, 2014. Revised Selected Papers*, pages 89–97, 2014.

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

[BBB13] Craig Boutilier, Fahiem Bacchus, and Ronen I. Brafman. Ucp-networks : A directed graphical representation of conditional utilities. *CoRR*, abs/1301.2259, 2013.

[BBHP99] Craig Boutilier, Ronen I. Brafman, Holger H. Hoos, and David Poole. Reasoning with conditional ceteris paribus preference statements. In *Proceedings of the Fifth conference on Uncertainty in Artificial Intelligence*, UAI'99, pages 71–80, 1999.

[BC00] T. Baccino and T. Colombi. L'analyse des mouvements des yeux sur le web. *Revue d'Intelligence Artificielle*, 14 :127–148, 2000.

[BHM14a] Yoann Bonavero, Marianne Huchard, and Michel Meynard. Improving Web Accessibility : Computing New Web Page Design with NSGA-II for People with Low Vision. *International Journal on Advances in Internet Technology, issn 1942-2652*, 7(3-4) :243–261, 2014.

[BHM14b] Yoann Bonavero, Marianne Huchard, and Michel Meynard. Web page personalization to improve e-accessibility for visually impaired people. In *Proceedings of the Second International Conference on Building and Exploring Web Based Environments (WEB 2014)*, pages 40–45, 2014.

[BHM15] Yoann Bonavero, Marianne Huchard, and Michel Meynard. Reconciling user and designer preferences in adapting web pages for people with low vision. In Luís Carriço, Silvia Mirri, Tiago João Guerreiro, and Peter Thiessen, editors, *Proceedings of the 12th Web for All Conference, W4A '15, Florence, Italy, May 18-20, 2015*, pages 10 :1–10 :10. ACM, 2015.

[BKL+06] Jeffrey P. Bigham, Ryan S. Kaminsky, Richard E. Ladner, Oscar M. Danielsson, and Gordon 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.

[BL07] Jeffrey P Bigham and Richard 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.

[Chi] Tsung-Che Chiang. *A NSGA-III non official C++ implementation*. `http://web.ntnu.edu.tw/~tcchiang/publications/nsga3cpp/nsga3cpp.htm`.

[CHS15] Hossein Rajabalipour Cheshmehgaz, Habibollah Haron, and Abdollah Sharifi. The review of multiple evolutionary searches and multi-objective evolutionary algorithms. *Artif. Intell. Rev.*, 43(3) :311–343, 2015.

[Col08] Sonia Colas. *Outils d'amélioration de l'accessibilité du Web pour les personnes visuellement handicapées*. PhD thesis, Université François Rabelais - Tours,, 2008.

[DAPM02a] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm : NSGA-II. *IEEE Trans. Evolutionary Computation*, 6(2) :182–197, 2002.

[DAPM02b] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm : NSGA-II. *IEEE Trans. Evolutionary Computation*, 6(2) :182–197, 2002.

[Dir09] Dir. Générale de la Modernisation de l'Etat (DGME), `http://references.modernisation.gouv.fr/rgaa-accessibilite`. *Référentiel Général d'Accessibilité pour les Administrations version 2.2.1*, 2009. Accessed : 2014-11-09.

[DJ14] Kalyanmoy Deb and Himanshu Jain. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I : solving problems with box constraints. *IEEE Trans. Evolutionary Computation*, 18(4) :577–601, 2014.

[FMPS14a] Stefano Ferretti, Silvia Mirri, Catia Prandi, and Paola 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.

[FMPS14b] Stefano Ferretti, Silvia Mirri, Catia Prandi, and Paola 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.

[FS09] Antonella Foti and Giuseppe Santucci. Increasing web accessibility through an assisted color specification interface for colorblind people. *IxD&A*, 5-6 :41–48, 2009.

[HH92] Peter Haddawy and Steve Hanks. Representations for decision-theoretic planning : Utility functions for deadline goals. In *KR 1992*, pages 71–82. Morgan Kaufmann, 1992.

[HMZ12] Mark Harman, S. Afshin Mansouri, and Yuanyuan Zhang. Search-based software engineering : trends, techniques and applications. *ACM Comput. Surv.*, 45(1) :11 :1–11 :61, December 2012.

[IBR10] Muhammad Asiful Islam, Yevgen Borodin, and I. V. Ramakrishnan. Mixture model based label association techniques for web accessibility. In Ken Perlin, Mary Czerwinski, and Rob Miller, editors, *UIST*, pages 67–76. ACM, 2010.

[JD14a] H. Jain and K. Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii : Handling constraints and extending to an adaptive approach. *Evolutionary Computation, IEEE Transactions on*, 18(4) :602–622, Aug 2014.

[JD14b] Himanshu Jain and Kalyanmoy Deb. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II : handling constraints and extending to an adaptive approach. *IEEE Trans. Evolutionary Computation*, 18(4) :602–622, 2014.

[Kac11] Souhila Kaci. *Working with Preferences : Less Is More*. Cognitive Technologies. Springer, 2011.

[LBH08] Darren Lunn, Sean Bechhofer, and Simon Harper. A user evaluation of the sadie transcoder. In Simon Harper and Armando 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.

[LHB09] Darren Lunn, Simon Harper, and Sean Bechhofer. Combining sadie and axsjax to improve the accessibility of web content. In David Sloan, Chieko Asakawa, and Hironobu 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.

[Lie09] Arnaud Liefooghe. *Metaheuristics for multiobjective optimisation : Cooperative approaches, uncertainty handling and application in logistics*. Theses, Université des Sciences et Technologie de Lille - Lille I, December 2009.

[LIR15] LIRIS. *Natbraille un transcripteur Braille libre*, Accessed : 2015-03-25 2015. `http://natbraille.free.fr/`.

[MAMS14] Alina Mereuta, Sebastien Aupetit, Nicolas Monmarché, and Mohamed Slimane. Web page textual color contrast compensation for CVD users using optimization methods. *J. Math. Model. Algorithms in OR*, 13(4) :447–470, 2014.

[MGS02] Mercedes Macías, Julia González, and Fernando Sánchez. On adaptability of Web sites for visually handicapped people. In Paul De Bra, Peter Brusilovsky, and Ricardo 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.

[MPS13] Silvia Mirri, Catia Prandi, and Paola 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.

[MSPM12] Silvia Mirri, Paola Salomoni, Catia Prandi, and Ludovico Antonio Muratori. Gapforape : an augmented browsing system to improve web 2.0 accessibility. *New Review of Hypermedia and Multimedia*, 18(3) :205–229, 2012.

[OMSa] OMS. *Maladies oculaires prioritaires - Cataracte*. `http://www.who.int/blindness/causes/priority/fr/index1.html`.

[OMSb] OMS. *Maladies oculaires prioritaires - Dégénérescence maculaire liée à l'âge (DMLA)*. `http://www.who.int/blindness/causes/priority/fr/index8.html`.

[OMSc] OMS. *Maladies oculaires prioritaires - Erreurs de réfraction*. `http://www.who.int/blindness/causes/priority/fr/index5.html`.

[OMSd] OMS. *Maladies oculaires prioritaires - Glaucome*. `http://www.who.int/blindness/causes/priority/fr/index7.html`.

[OMSe] OMS. *Maladies oculaires prioritaires - Rétinopathies diabétiques*. `http://www.who.int/blindness/causes/priority/fr/index6.html`.

[OMSf] *Rapport mondial sur le handicap 2011*. `http://www.who.int/disabilities/world_report/2011/report/fr/index.html`.

[Pet14] Franck Petitdemange. Inférence de la structure d'une page web en vue d'améliorer son accessibilité. Master's thesis, Université Montpellier 2, Sciences et Techniques du Languedoc, juin 2014.

[Pib15] Lionel Pibre. Stéganalyse par "deep learning". Master's thesis, Université Montpellier, juin 2015.

[RDK08] Vikas C. Raykar, Ramani Duraiswami, and Balaji Krishnapuram. A fast algorithm for learning a ranking function from large-scale data sets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(7) :1158–1170, 2008.

[San09] Giuseppe Santucci. Vis-a-wis : Improving visual accessibility through automatic web content adaptation. In Constantine 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.

[TBM08] Luigi Troiano, Cosimo Birtolo, and Maria 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.

[TCH+02] (Beth Tibbitts, Susan Crayne, Vicki Hanson, Jonathan Brezin, Cal Swart, and John Richards. HTML parsing in Java for accessibility transformation. In *Proceedings of XML 2002 – XML Conference and Exposition*, 2002.

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

[VW63] Georg Henrik Von Wright. *The logic of Preference : an Essay*. University Press, 1963.

[W3C15] W3C. *Contrast (Minimum) : Understanding SC 1.4.3*, Accessed : 2015-03-25 2015. `http://www.`

w3.org/TR/2008/WD-UNDERSTANDING-WCAG20-20080430/visual-audio-contrast-contrast.html.

[WCA] World Wide Web Consortium, http://www.w3.org/TR/WCAG20/. *Web Content Accessibility Guidelines*. Accessed : 2014-11-09.

[Web] *Web Accessibility Initiative*. http://www.w3.org/WAI/.

[wik15] wikipedia. *Contraste simultané*, Accessed : 2015-03-25 2015. https://fr.wikipedia.org/wiki/Loi_du_contraste_simultané_des_couleurs.

Annexe A

Pages Web pour l'expérimentation

Préambule

Cette première annexe regroupe l'ensemble des pages Web réelles qui ont servi pour réaliser les expérimentations. Ces pages n'ont pas été prises complètement au hasard. Elles répondent à certains critères comme la diversité du nombre d'éléments affichés et une certaine susceptibilité à engendrer des difficultés d'ordre visuel. Dix pages de sites Web différents ont été ainsi sélectionnées. Quelques problèmes potentiels sont également mis en avant pour chacune de ces pages Web. Le détail des problèmes de visualisation pouvant être rencontrés se trouve dans le chapitre 4.

A.1 Legibase (Berger-Levrault)



FIGURE A.1 – Page du site Web *Legibase*

Le site Web Legibase fait partie des trois sites que nous avons utilisés et qui ont été fournis par l'entreprise Berger-Levrault qui a soutenu ces travaux de recherches. Il s'agit ici d'une page de type contenu avec peu d'éléments dynamiques. Dans l'ensemble, elle contient assez d'éléments différents. Des difficultés peuvent intervenir lors de la visualisation de cette page, certainement autour de l'encart en gris du bas de la page. En effet, le contraste entre le texte et l'arrière-plan est assez réduit. De plus, cet encart dispose d'un arrière-plan plutôt sombre alors que le reste de la page a un arrière-plan lumineux. Si un utilisateur utilise des filtres de couleurs (fournis par exemple par son magnifier d'écran) il peut ne pas arriver à adapter la page dans son intégralité.

A.2 National Federation of the Blind (NFB)

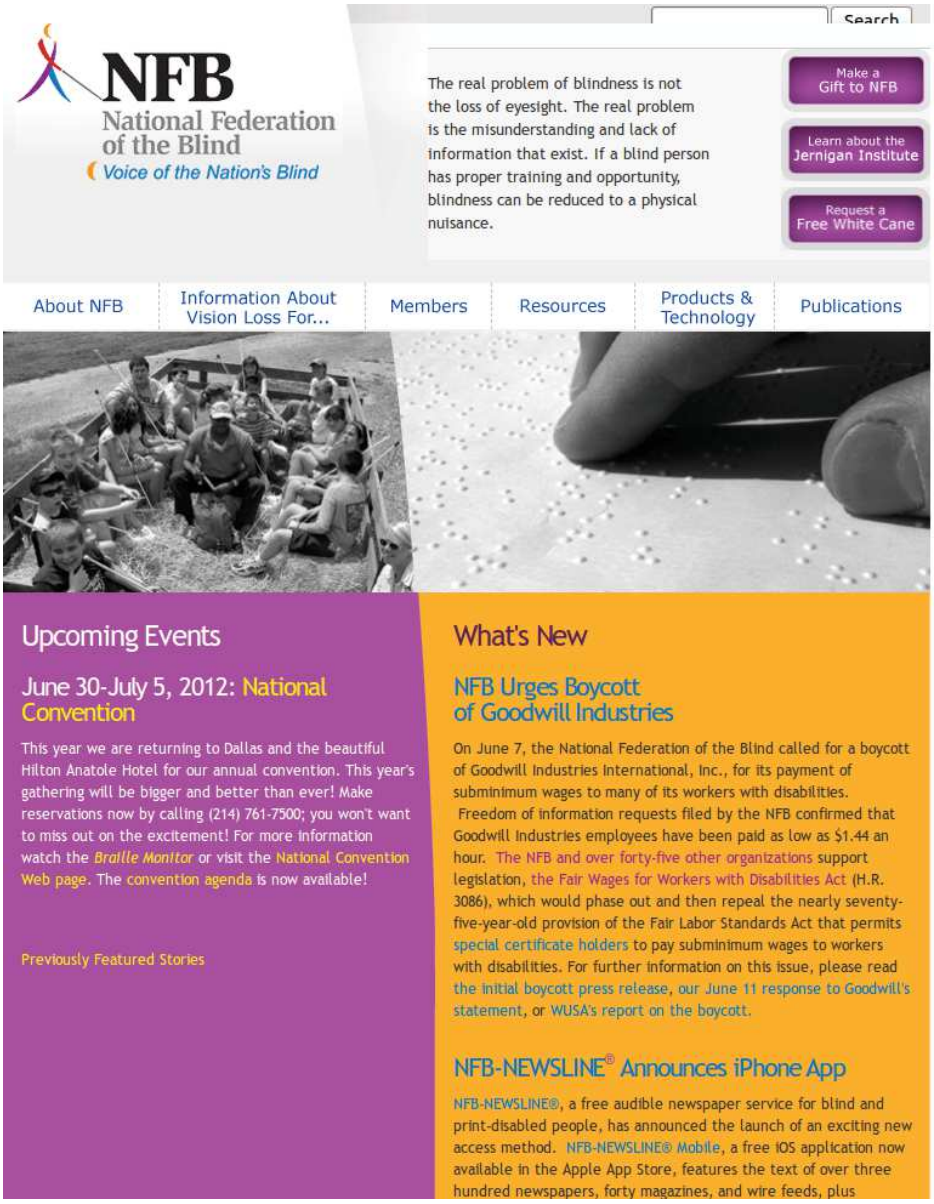


FIGURE A.2 – Page du site Web NFB

URL : `http://www.nfb.org`

La page du site Web de la National Federation of the Blind, bien que destinée en grande partie à des personnes déficientes visuelles, possède un certain nombre d'éléments pouvant poser des problèmes. Par exemple, les deux grandes sections de texte se trouvant sur la gauche sur fond violet et sur la droite sur fond orangé ont un contraste plutôt faible. En effet, certaines personnes sont incapables de lire le contenu textuel tel qu'il est affiché. Les boutons se trouvant en haut à droite peuvent également poser problème dans le sens où il sont insérés en tant qu'images contenant du texte et non en tant que texte avec une image à l'arrière.

A.3 GDArchitect

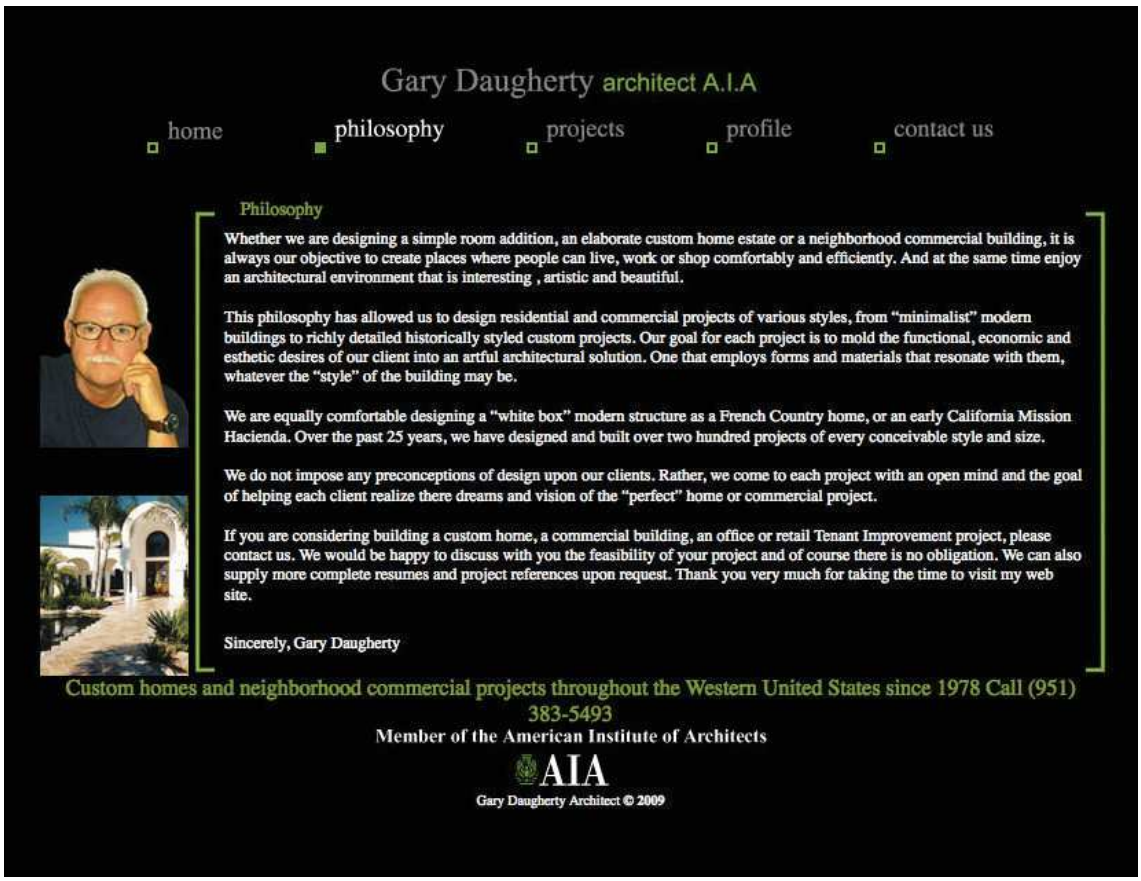


FIGURE A.3 – Page du site Web *GDArchitect*

URL : `http://gdarchitect.com/philosophy.htm`

Il s'agit ici d'une page sur fond sombre avec un texte en général relativement clair. Cette page, issue du site Web GDArchitect, est une page de contenu à tendance fixe. Cette page décrit leur philosophie et a de fortes chances de rester ainsi longtemps. Les standards et recommandations d'accessibilité semblent être respectés. Il se peut que cette page ne pose que peu de problèmes aux personnes ayant une basse vision. Cependant, pour certaines personnes, le fond sombre peut être un problème, puisque ce dernier fait opposition aux fonds très souvent clairs par défaut des menus et widgets du navigateur.

Si cette page est dans l'ensemble assez satisfaisante sur plusieurs points, notamment le respect d'un grand nombre de standards d'accessibilité, elle reste néanmoins utile pour les expérimentations. En effet, utiliser une page pour laquelle les souhaits de l'utilisateur sont presque déjà satisfaits permet de voir l'impact de la recherche d'une solution sur cette page et d'observer les potentielles dégradations.

A.4 My Account (Berger-Levrault)

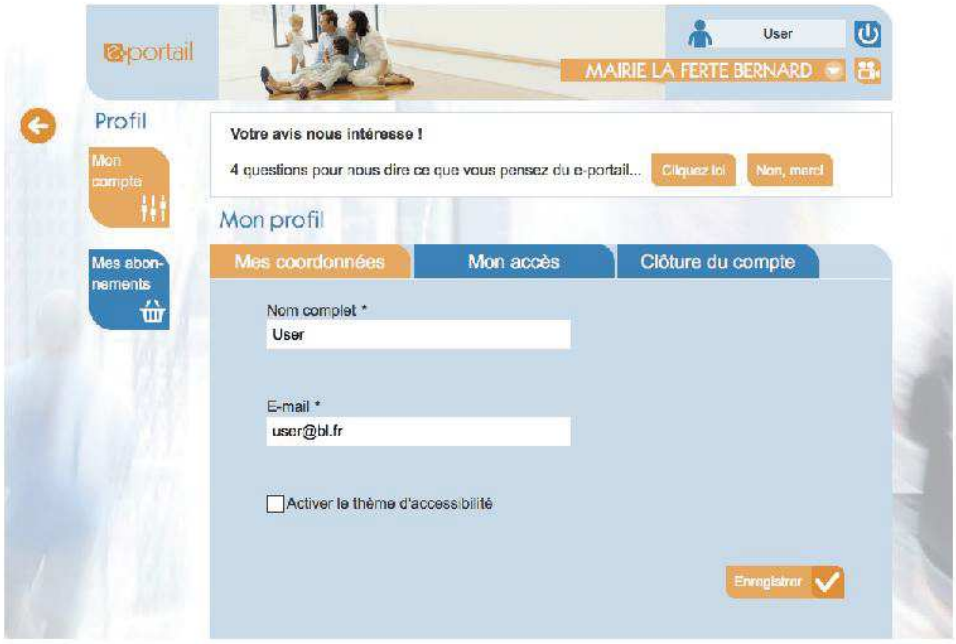


FIGURE A.4 – Page du site Web de Berger-Levrault

Cette page Web est tirée d'un des sites Web de Berger-Levrault encore en développement. Pour des raisons de confidentialité, l'url de cette page n'est pas communiquée. Il s'agit ici d'une page un peu différente des autres dans le sens où elle offre des éléments d'interaction permettant de se connecter à son compte. La page est assez simple visuellement. Les difficultés peuvent arriver principalement sur les boutons d'interaction. En effet, certains d'entre eux sont insérés par le biais d'images. De plus, les contrastes entre le texte et l'arrière-plan de ces boutons ne sont pas forcément très élevés. On peut également imaginer que l'arrière-plan se trouvant sous le formulaire d'authentification peut poser problème selon le style d'affichage du texte se trouvant au-dessus.

A.5 Parempuyre



FIGURE A.5 – Page du site Web Parempuyre

URL : `http://www.parempuyre.fr`

Le site Web de la ville de Parempuyre utilise plusieurs images et animations pour décorer ses pages. Les vignettes rondes sur la gauche peuvent faire penser à des boutons mais ce n'en est pas. La colonne principale où se trouve le contenu de la page a un arrière-plan dans les tons de bleu gris assez clair. Le texte qui se trouve au-dessus est en majorité blanc. Le contraste entre le contenu textuel et l'arrière-plan est donc relativement faible ici. Le niveau de contraste entre le texte et l'arrière-plan semble être la principale difficulté sur cette page.

A.6 Godaddy



FIGURE A.6 – Page du site Web *Godaddy*

URL : `http://fr.godaddy.com/tlds/co-domain.aspx`

Cette page est constituée d'un grand nombre d'éléments variés. Elle est également assez colorée. Par exemple la barre d'en-tête et le pied de page ont un arrière-plan gris foncé (sombre) tandis que le reste de la page a un arrière-plan blanc (clair). La barre juste avant le pied de page et les menus déroulants ont un arrière-plan vert avec un texte blanc. De nombreux problèmes de contrastes peuvent arriver sur cette page du fait notamment d'un grand nombre de couleurs utilisées. Le mélange de couleurs sombres et lumineuses pour les arrière-plans peut également causer des difficultés de lecture.

A.7 FamilyPortal (Berger-Levrault)



FIGURE A.7 – Page du site Web de Berger-Levrault *Portail famille*

Cette nouvelle page fournie par Berger-Levrault montre un aperçu de l'affichage des factures pour une compte utilisateur. Le portail famille est en cours de développement par l'entreprise, donc par souci de confidentialité l'URL ne peut être donnée. Sur cette page, les boutons d'interaction peuvent être à l'origine de problèmes du fait d'insertion sous la forme d'images (notamment on ne peut pas les adapter). Par exemple, dans le tableau des factures, le symbole « € » (euro) est inséré sous la forme d'une image et n'est par conséquent pas du texte. Sur les boutons de gauche le contraste entre le texte blanc et le fond bleu n'est pas forcément très suffisant pour tout le monde. De la même manière, le contraste entre le texte et l'arrière-plan de l'en-tête et du pied du tableau est aussi relativement faible. De manière générale, le thème d'écriture blanc sur le fond bleu choisi peut véritablement être sujet à des difficultés de lecture. Pour finir, le petit titre (principal) où est écrit « Portail » a une couleur de caractères (dans les jaunes) qui peut réellement poser des problèmes de lecture par rapport à son arrière-plan très clair.

A.8 UBOFormation



FIGURE A.8 – Page du site Web UBO Formations

URL : http://formations.univ-brest.fr/fiche/FR_RNE_0290346U_PROG17269/FR_RNE_0290346U_PROG37086/presentation

Ce site de formation/stages possède beaucoup d'éléments sur ses pages. La page d'exemple que nous avons choisie contient beaucoup de textes dans les tons de gris. De plus, ses textes sont régulièrement peu contrastés par rapport à leur arrière-plan. La distinction entre les liens et le texte (régulier) n'est pas évidente, il n'existe pas vraiment de différence de couleur entre les deux et les liens ne sont pas soulignés. Par exemple dans la partie de droite dans la section « contacts », l'adresse mail est un lien alors que le numéro de téléphone n'en est pas un.

A.9 UBOLEA

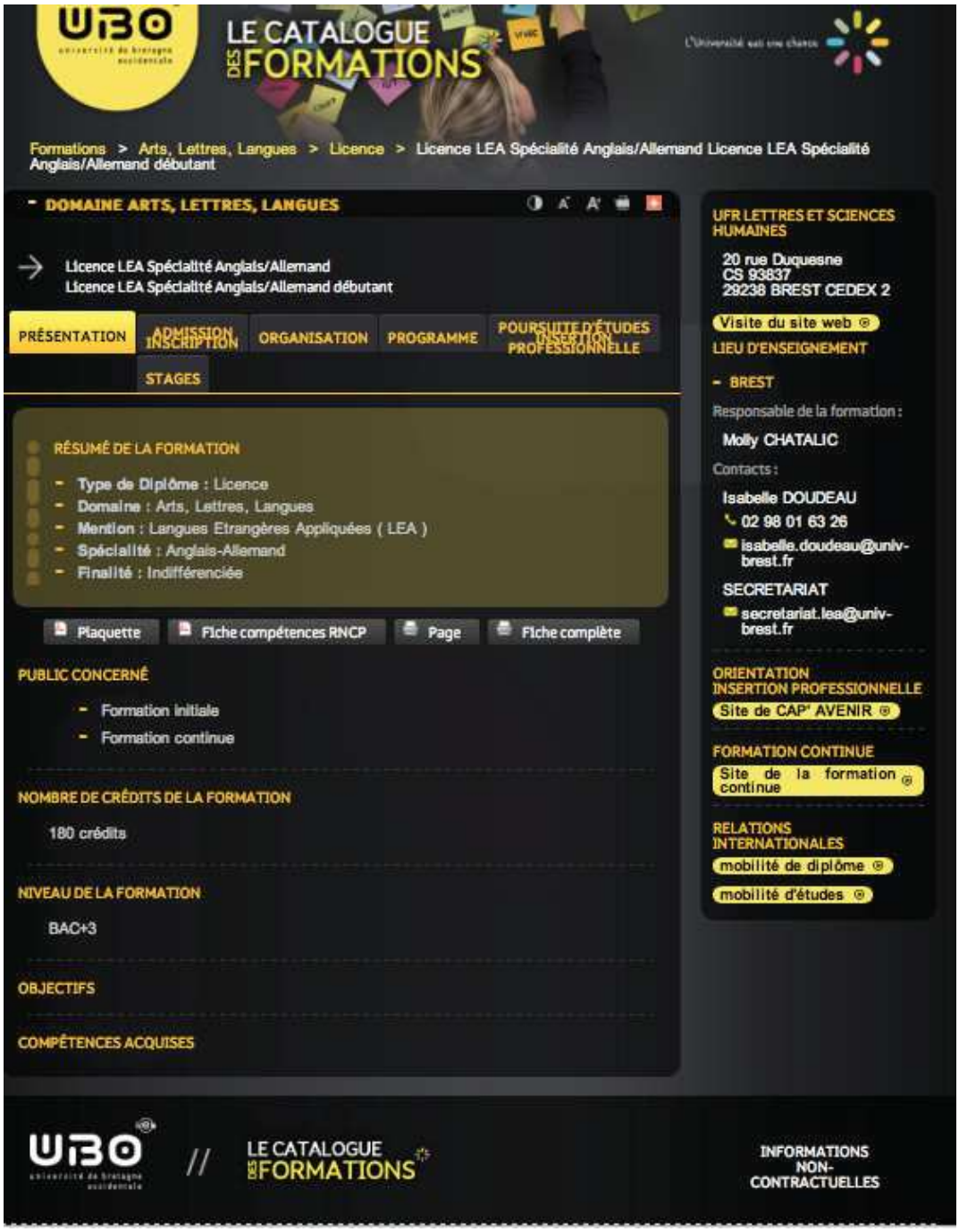


FIGURE A.9 – Page du site Web UBO Licence LEA

URL : http://formations.univ-brest.fr/fiche/FR_RNE_0290346U_PROG17269/FR_RNE_0290346U_PROG37086/presentation

Il s'agit ici d'une autre page du site Web UBO. Cette page concerne la licence LEA. Elle présente plus ou moins les même propriétés que la page précédente. De ce fait, les difficultés pouvant être rencontrées seront relativement similaires.

A.10 ReasonFrance

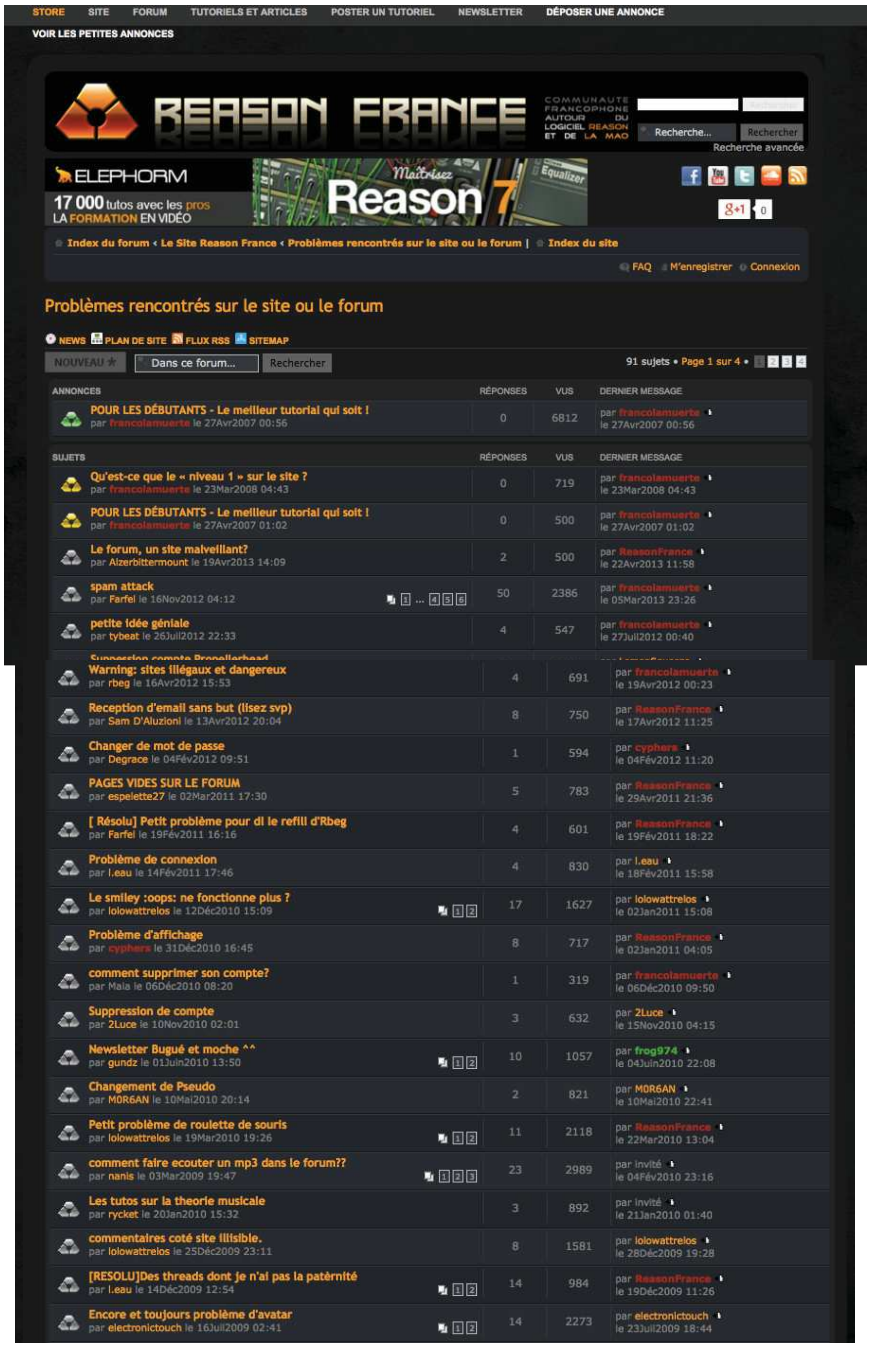


FIGURE A.10 – Page du site Web Reason France (forum)

URL : `http://forum.reasonfrance.fr`

Le forum ReasonFrance est l'exemple le plus gros de notre échantillon. Il compte un grand nombre d'éléments textuels et d'arrière-plans. Comme sur la plupart des autres exemples, on peut rencontrer des difficultés par rapport au contraste texte - arrière-plan. Par exemple le nom (pseudo) de la personne qui a déposé le message ou ouvert un sujet n'est pas très contrasté avec l'arrière-plan d'un point de vue de la brillance de la couleur.

Annexe B

Extraction des éléments des pages Web

Préambule

Cette annexe contient l'ensemble des fichiers créés et utilisés afin de réaliser la sélection des éléments depuis les pages. Les éléments sont sélectionnés en fonction des objectifs de modification définis. Pour chaque page du corpus de pages web utilisé, nous avons d'un côté, les images créées contenant les noms des variables avec une indication de l'élément auquel elles se rapportent. D'un autre côté, nous avons le contenu du fichier où ces variables ainsi que la déclaration de tous les objectifs sont stockées.

B.1 Legibase (Berger-Levrault)

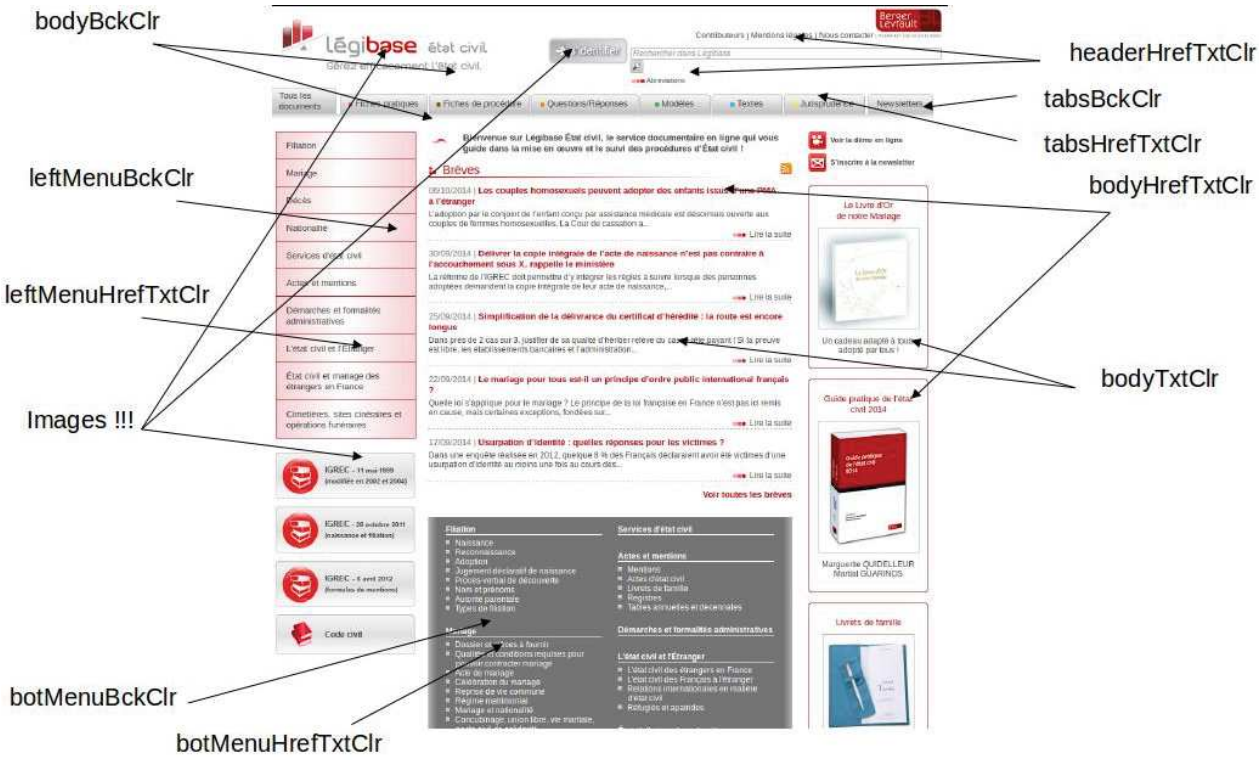


FIGURE B.1 – Page du site Web *Legibase*

Variables :

0 - bodyBckClr : 255 255 255	*1- leftMenuBckClr : 248 218 222 (image dégradé)*
2- botMenuBckClr : 116 116 116	*3- headerHrefTxtClr : 70 70 70*
4- bodyHrefTxtClr : 186 19 25	*5- bodyTxtClr : 70 70 70*
6- leftMenuHrefTxtClr : 70 70 70	*7- botMenuHrefTxtClr : 255 255 255*

Contraintes de contraste de brillance :

$contrast(0,\ 3) \geq x$	$contrast(0,\ 4) \geq x$
$contrast(0,\ 5) \geq x$	$contrast(1,\ 6) \geq x$
$contrast(2,\ 7) \geq x$	

Contrainte de proximité de brillance d'arrière-plans :

proxBrightness(0,1,2)

Contraintes de distance de couleur :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$
$distOrigClr(2) \leq y$	$distOrigClr(3) \leq y$
$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$
$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$

B.2 National Federation of the Blind (NFB)

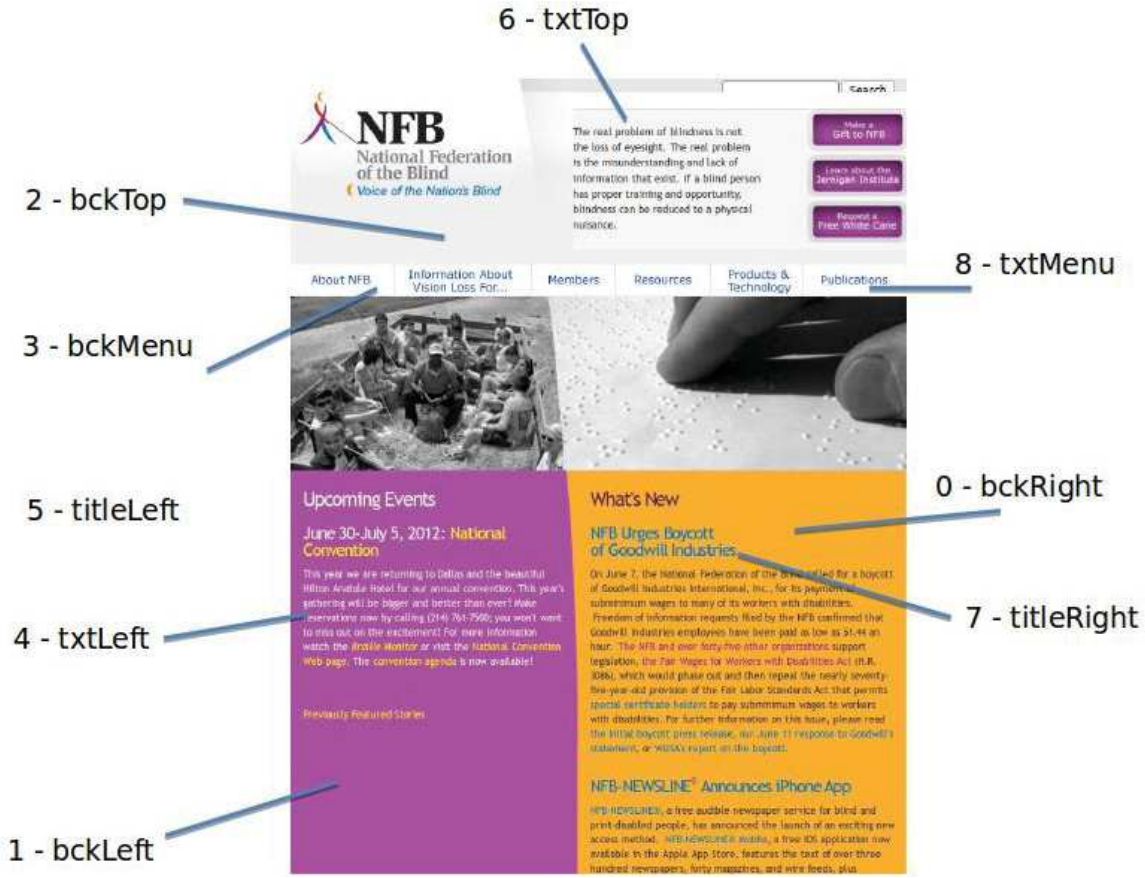


FIGURE B.2 – Page du site Web NFB

Variables :

0 - bckRight : 225 133 1	*1- bckLeft : 145 60 141*
2- bckTop : 231 231 231	*3- bckMenu : 255 255 255*
4- txtLeft : 255 255 255	*5- titleLeft : 255 255 255*
6- txtTop : 0 0 0	*7- titleRight : 31 102 195*
8- txtMenu : 31 102 195	

Contraintes de contraste de brillance :

$contrast(0,\ 4) \geq x$	$contrast(1,\ 5) \geq x$	$contrast(2,\ 6) \geq x$
$contrast(3,\ 7) \geq x$	$contrast(3,\ 8) \geq x$	

Contrainte de proximité de brillance d'arrière-plans :

proxBrightness(0,1,2,3)

Contraintes de distance de couleur :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$	$distOrigClr(2) \leq y$
$distOrigClr(3) \leq y$	$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$
$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$	$distOrigClr(8) \leq y$

B.3 GDArchitect

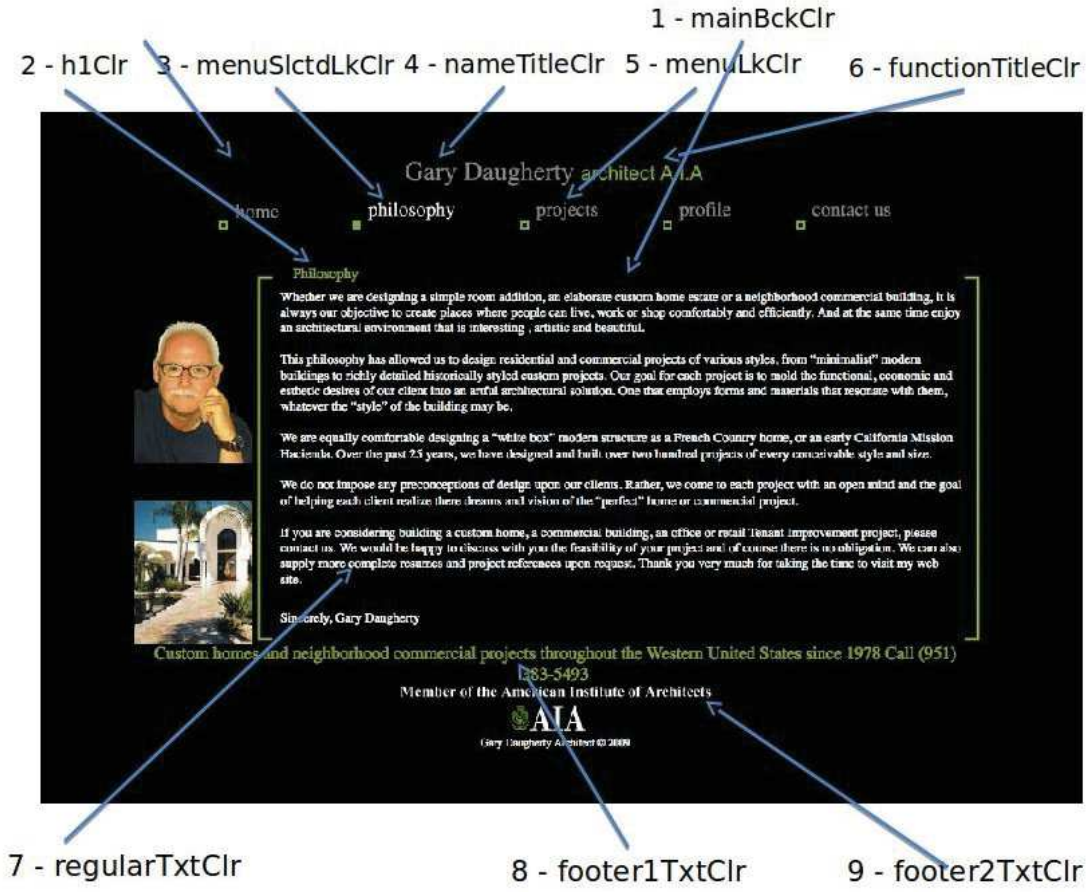


FIGURE B.3 – Page du site Web *GDArchitect*

Variables :

0 - pageBckClr : 000, 000, 000	*1 - mainBckClr : 000, 000, 000*
2 - h1Clr : 106 151 55	*3 - menuSlctdLkClr : 255, 255, 255*
4 - nameTitleClr : 122 122 122	*5 - menuLkClr : 122 122 122*
6 - functionTitleClr : 106 151 55	*7 - regularTxtClr : 255, 255, 255*
8 - footer1TxtClr : 106 151 55	*9 - footer2TxtClr : 255, 255, 255*

Contraintes de contraste de brillance :

$contrast(2,\ 0) \geq x$	$contrast(3,\ 0) \geq x$	$contrast(4,\ 0) \geq x$
$contrast(5,\ 0) \geq x$	$contrast(6,\ 0) \geq x$	$contrast(7,\ 1) \geq x$
$contrast(8,\ 0) \geq x$	$contrast(9,\ 0) \geq x$	

Contrainte de proximité de brillance d'arrière-plans :

proxBrightness(0,1)

Contraintes de distance de couleurs :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$	$distOrigClr(2) \leq y$
$distOrigClr(3) \leq y$	$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$
$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$	$distOrigClr(8) \leq y$
$distOrigClr(9) \leq y$		

B.4 My Account (Berger-Levrault)



FIGURE B.4 – Page du site Web de Berger-Levrault

Variables :

0 - bodyBckClr 233 239 244	*1 - fieldBckClr 228 236 244*
2 - menuBckClr 224 153 80	*3 - boiteblancheBckClr 255 255 255*
4 - orgButBckClr 224 153 80	*5 - orgOngletBckClr 224 153 80*
6 - blueOngletBckClr 49 109 167	*7 - boitebleueBckClr 190 209 226*
8 - fieldblancBckClr 255 255 255	*9 - textblueClr 54 109 168*
10 - fieldTxtClr 40 40 40	*11 - menuTxtClr 255 255 255*
12 - boiteblancheTxtClr 40 40 40	*13 - orgButTxtClr 255 255 255*
14 - orgOngletTxtClr 255 255 255	*15 - blueOngletTxtClr 255 255 255*
16 - boitebleueTxtClr 40 40 40	*17 - fieldblancTxtClr 0 0 0*

Contraintes de contraste de brillance :

$contrast(0,\ 9) \geq x$	$contrast(1,\ 10) \geq x$	$contrast(2,\ 11) \geq x$
$contrast(3,\ 12) \geq x$	$contrast(4,\ 13) \geq x$	$contrast(5,\ 14) \geq x$
$contrast(6,\ 15) \geq x$	$contrast(7,\ 16) \geq x$	$contrast(8,\ 17) \geq x$

Contrainte de proximité de brillance d'arrière-plans :

proxBrightness(0,1,2,3,4,5,6,7,8)

Contraintes de distance de couleurs :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$	$distOrigClr(2) \leq y$	$distOrigClr(3) \leq y$
$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$	$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$
$distOrigClr(8) \leq y$	$distOrigClr(9) \leq y$	$distOrigClr(10) \leq y$	$distOrigClr(11) \leq y$
$distOrigClr(12) \leq y$	$distOrigClr(13) \leq y$	$distOrigClr(14) \leq y$	$distOrigClr(15) \leq y$
$distOrigClr(16) \leq y$	$distOrigClr(17) \leq y$		

B.5 Parempuyre

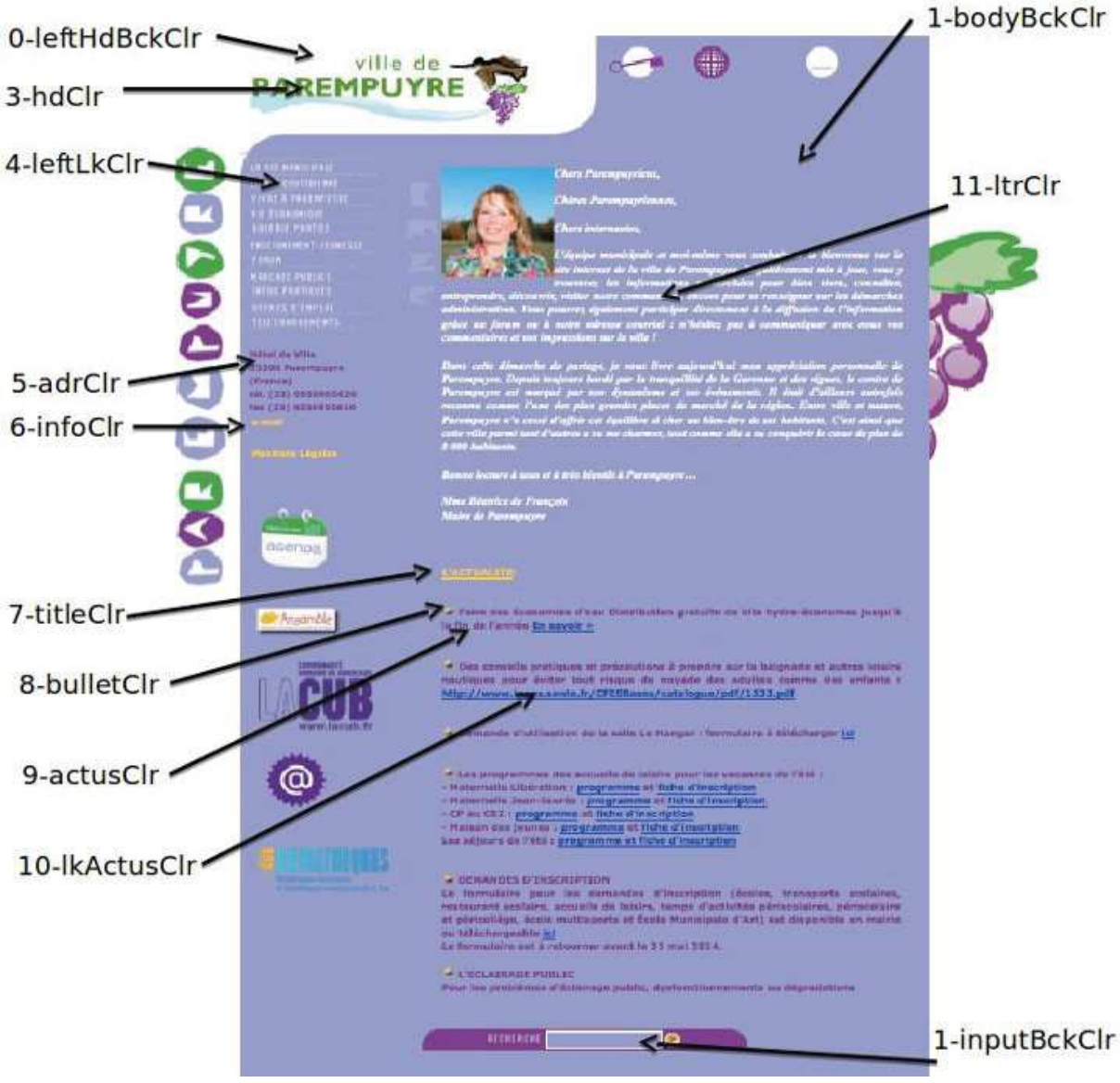


FIGURE B.5 – Page du site Web

Variables :

0 - leftHdBckClr : 255 255 255	*1 - bodyBckClr : 133 138 190*	*2 - inputBckClr : 133 138 190*
3 - hdClr : 70 136 35	*4 - leftLkClr : 255 255 255*	*5 - adrClr : 105 42 124*
6 - infoClr : 248 192 57	*7 - titleClr : 248 192 57*	*8 - bulletClr : 164 88 28*
9 - actusClr : 105 42 124	*10 - lkActusClr : 0 37 229*	*11 - ltrClr : 255 255 255*

Contraintes de contraste de brillance :

$contrast(0,\ 3) \geq x$	$contrast(1,\ 4) \geq x$	$contrast(1,\ 5) \geq x$
$contrast(1,\ 6) \geq x$	$contrast(1,\ 7) \geq x$	$contrast(1,\ 8) \geq x$
$contrast(1,\ 9) \geq x$	$contrast(1,\ 10) \geq x$	$contrast(1,\ 11) \geq x$

Contrainte de proximité de brillance d'arrière-plans :

- *proxBrightness(0,1,2)*

Contraintes de distance de couleurs :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$	$distOrigClr(2) \leq y$	$distOrigClr(3) \leq y$
$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$	$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$
$distOrigClr(8) \leq y$	$distOrigClr(9) \leq y$	$distOrigClr(10) \leq y$	$distOrigClr(11) \leq y$

B.6 Godaddy

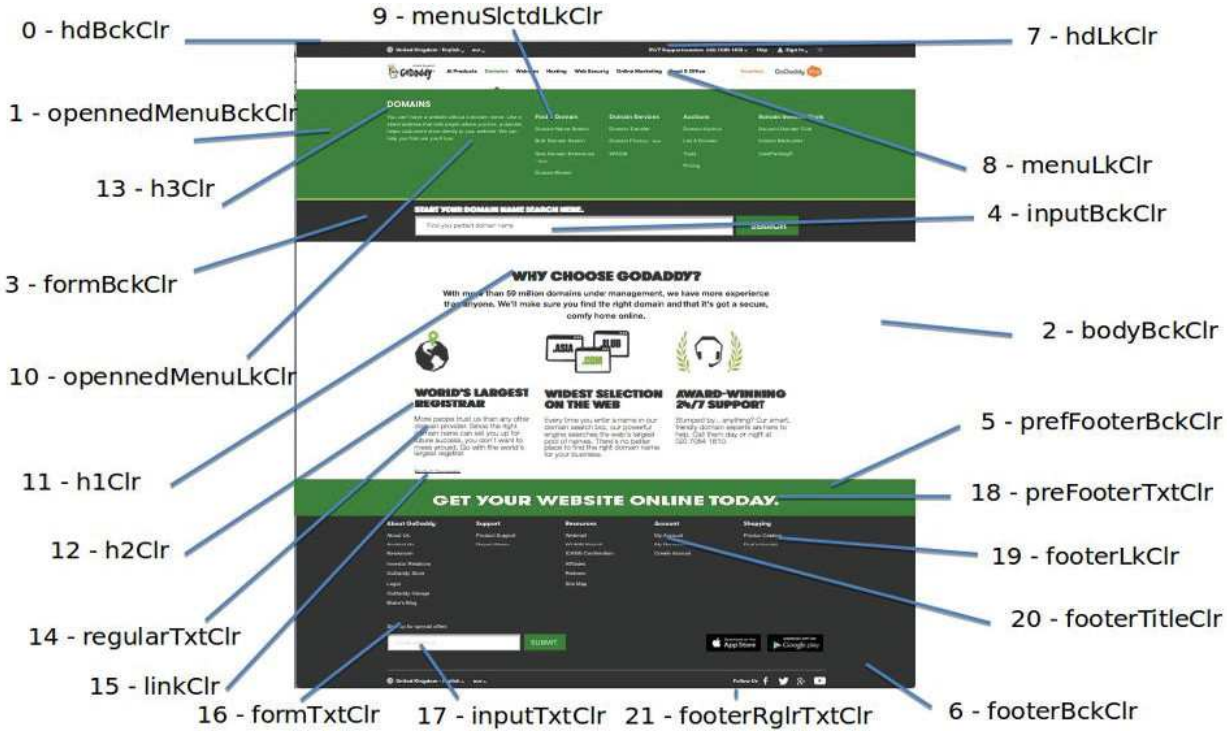


FIGURE B.6 – Page du site Web *Godaddy*

Variables :

0 - hdBckClr : 051, 051, 051	*1 - opennedMenuBckClr : 105, 158, 003*
2 - bodyBckClr : 255, 255, 255	*3 - formBckClr : 051, 051, 051*
4 - inputBckClr : 255, 255, 255	*5 - preFooterBckClr : 104, 154, 029*
6 - footerBckClr : 051, 051, 051	*7 - hdLkClr : 255, 255, 255*
8 - menuLkClr : 000, 000, 000	*9 - menuSlctdLkClr : 125, 183, 001*
10 - opennedMenuLkClr : 255, 255, 255	*11 - h1Clr : 000, 000, 000*
12 - h2Clr : 000, 000, 000	*13 - h3Clr : 000, 000, 000*
14 - regularTxtClr : 000, 000, 000	*15 - linkClr : 113, 113, 113*
16 - formTxtClr : 255, 255, 255	*17 - inputTxtClr : 000, 000, 000*
18 - preFooterTxtClr : 255, 255, 255	*19 - footerLkClr : 212, 255, 255*
20 - footerTitleClr : 255, 255, 255	*21 - footerRglrTxtClr : 255, 255, 255*

Contraintes de contraste de brillance :

$contrast(0,\ 7) \geq x$	$contrast(1,\ 10) \geq x$	$contrast(2,\ 8) \geq x$	$contrast(2,\ 11) \geq x$
$contrast(2,\ 12) \geq x$	$contrast(2,\ 13) \geq x$	$contrast(2,\ 14) \geq x$	$contrast(2,\ 15) \geq x$
$contrast(3,\ 16) \geq x$	$contrast(4,\ 17) \geq x$	$contrast(5,\ 18) \geq x$	$contrast(6,\ 19) \geq x$
$contrast(6,\ 20) \geq x$	$contrast(6,\ 21) \geq x$		

Contrainte de proximité de brillance d'arrière-plans :

proxBrightness(0,1,2,3,4,5,6)

Contraintes de distance de couleurs :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$	$distOrigClr(2) \leq y$	$distOrigClr(3) \leq y$
$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$	$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$
$distOrigClr(8) \leq y$	$distOrigClr(9) \leq y$	$distOrigClr(10) \leq y$	$distOrigClr(11) \leq y$
$distOrigClr(12) \leq y$	$distOrigClr(13) \leq y$	$distOrigClr(14) \leq y$	$distOrigClr(15) \leq y$
$distOrigClr(16) \leq y$	$distOrigClr(17) \leq y$	$distOrigClr(18) \leq y$	$distOrigClr(19) \leq y$
$distOrigClr(20) \leq y$	$distOrigClr(21) \leq y$		

B.7 FamilyPortal (Berger-Levrault)

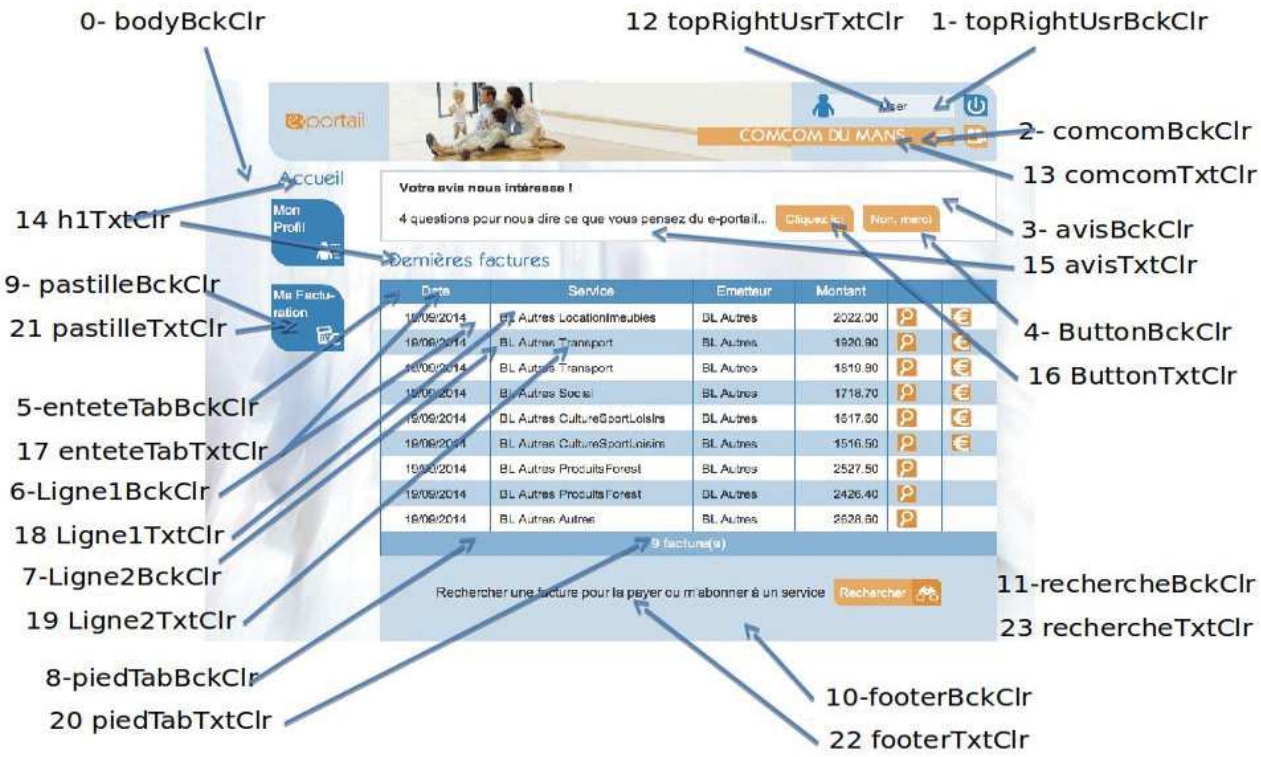


FIGURE B.7 – Page du site Web de Berger-Levrault *Portail famille*

Variables :

0- bodyBckClr : 233 237 243	*1- topRightUsrBckClr : 228 236 244*
2- comcomBckClr : 224 153 80	*3- avisBckClr : 255 255 255*
4- ButtonBckClr : 224 153 80	*5-enteteTabBckClr : 72 129 181*
6-Ligne1BckClr : 255 255 255	*7-Ligne2BckClr : 172 202 124*
8-piedTabBckClr : 118 163 202	*9- pastilleBckClr : 49 109 167*
10-footerBckClr : 190 209 226	*11-rechercheBckClr : 224 153 80*
12- topRightUsrTxtClr : 40 40 40	*13 - comcomTxtClr : 255 255 255*
14 h1TxtClr : 48 109 169	*15 avisTxtClr : 40 40 40*
16 ButtonTxtClr : 255 255 255	*17 enteteTabTxtClr : 255 255 255*
18 Ligne1TxtClr : 40 40 40	*19 Ligne2TxtClr : 40 40 40*
20 piedTabTxtClr : 255 255 255	*21 pastilleTxtClr : 255 255 255*
22 footerTxtClr : 40 40 40	*23 rechercheTxtClr : 255 255 255*

Contraintes de contraste de brillance :

$contrast(0,\ 14) \geq x$	$contrast(1,\ 12) \geq x$	$contrast(2,\ 13) \geq x$	$contrast(3,\ 15) \geq x$
$contrast(4,\ 16) \geq x$	$contrast(5,\ 17) \geq x$	$contrast(6,\ 18) \geq x$	$contrast(7,\ 19) \geq x$
$contrast(8,\ 20) \geq x$	$contrast(9,\ 21) \geq x$	$contrast(10,\ 22) \geq x$	$contrast(11,\ 23) \geq x$

Contrainte de proximité de brillance d'arrière-plans :

- *proxBrightness(0,1,2,3,4,5,6,7,8,9,10,11)*

Contraintes de distance de couleurs :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$	$distOrigClr(2) \leq y$	$distOrigClr(3) \leq y$
$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$	$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$
$distOrigClr(8) \leq y$	$distOrigClr(9) \leq y$	$distOrigClr(10) \leq y$	$distOrigClr(11) \leq y$
$distOrigClr(12) \leq y$	$distOrigClr(13) \leq y$	$distOrigClr(14) \leq y$	$distOrigClr(15) \leq y$
$distOrigClr(16) \leq y$	$distOrigClr(17) \leq y$	$distOrigClr(18) \leq y$	$distOrigClr(19) \leq y$
$distOrigClr(20) \leq y$	$distOrigClr(21) \leq y$	$distOrigClr(22) \leq y$	$distOrigClr(23) \leq y$

B.8 UBOFormation



FIGURE B.8 – Page du site Web UBO Formations

Variables :

0 - topBck : 38 39 42	*1 - menuPgBck : 36 36 39*
2 - itemPgBck : 178 114 152	*3 - buttonBck : 57 57 57*
4 - blockCtrBck : 20 20 20	*5 - blockRgtBck : 11 11 12*
6 - buttonRgtBck : 245 228 87	*7 - bottomPgBck : 16 16 17*
8 - h1l1Clr : 255 255 255	*9 - h1l2Clr : 246 226 80*
10 - itemClr : 166 102 132	*11 - itemPgClr : 000 000 000*
12 - buttonTxtClr : 255 255 255	*13 - h4Clr : 166 102 132*
14 - mainTxtClr : 255 255 255	*15 - bottTxt1Clr : 255 255 255*
16 - bottTxt2Clr : 246 226 80	*17 - fil1Clr : 246 226 80*
18 - fil2Clr : 255 255 255	*19 - h3Clr : 227 167 44*
20 - blockTxt1Clr : 166 103 140	*21 - blockTxt2Clr : 135 135 135*
22 - blockTxt3Clr : 255 255 255	*23 - buttonTxtClr : 000 000 000*

Contraintes de contraste de brillance :

contrast(0, 8) $\geq x$	*contrast(0, 9)* $\geq x$	*contrast(0, 17)* $\geq x$	*contrast(0, 18)* $\geq x$
contrast(1, 10) $\geq x$	*contrast(2, 11)* $\geq x$	*contrast(3, 12)* $\geq x$	*contrast(4, 13)* $\geq x$
contrast(4, 14) $\geq x$	*contrast(4, 19)* $\geq x$	*contrast(7, 15)* $\geq x$	*contrast(7, 16)* $\geq x$
contrast(5, 20) $\geq x$	*contrast(5, 21)* $\geq x$	*contrast(5, 22)* $\geq x$	*contrast(5, 6)* $\geq x$
contrast(6, 23) $\geq x$			

Contrainte de proximité de brillance d'arrière-plans :

proxBrightness(0,1,2,3,4,5,7)

Contraintes de distance de couleurs :

distOrigClr(0) $\leq y$	*distOrigClr(1)* $\leq y$	*distOrigClr(2)* $\leq y$	*distOrigClr(3)* $\leq y$
distOrigClr(4) $\leq y$	*distOrigClr(5)* $\leq y$	*distOrigClr(6)* $\leq y$	*distOrigClr(7)* $\leq y$
distOrigClr(8) $\leq y$	*distOrigClr(9)* $\leq y$	*distOrigClr(10)* $\leq y$	*distOrigClr(11)* $\leq y$
distOrigClr(12) $\leq y$	*distOrigClr(13)* $\leq y$	*distOrigClr(14)* $\leq y$	*distOrigClr(15)* $\leq y$
distOrigClr(16) $\leq y$	*distOrigClr(17)* $\leq y$	*distOrigClr(18)* $\leq y$	*distOrigClr(19)* $\leq y$
distOrigClr(20) $\leq y$	*distOrigClr(21)* $\leq y$	*distOrigClr(22)* $\leq y$	*distOrigClr(23)* $\leq y$

B.9 UBOLEA

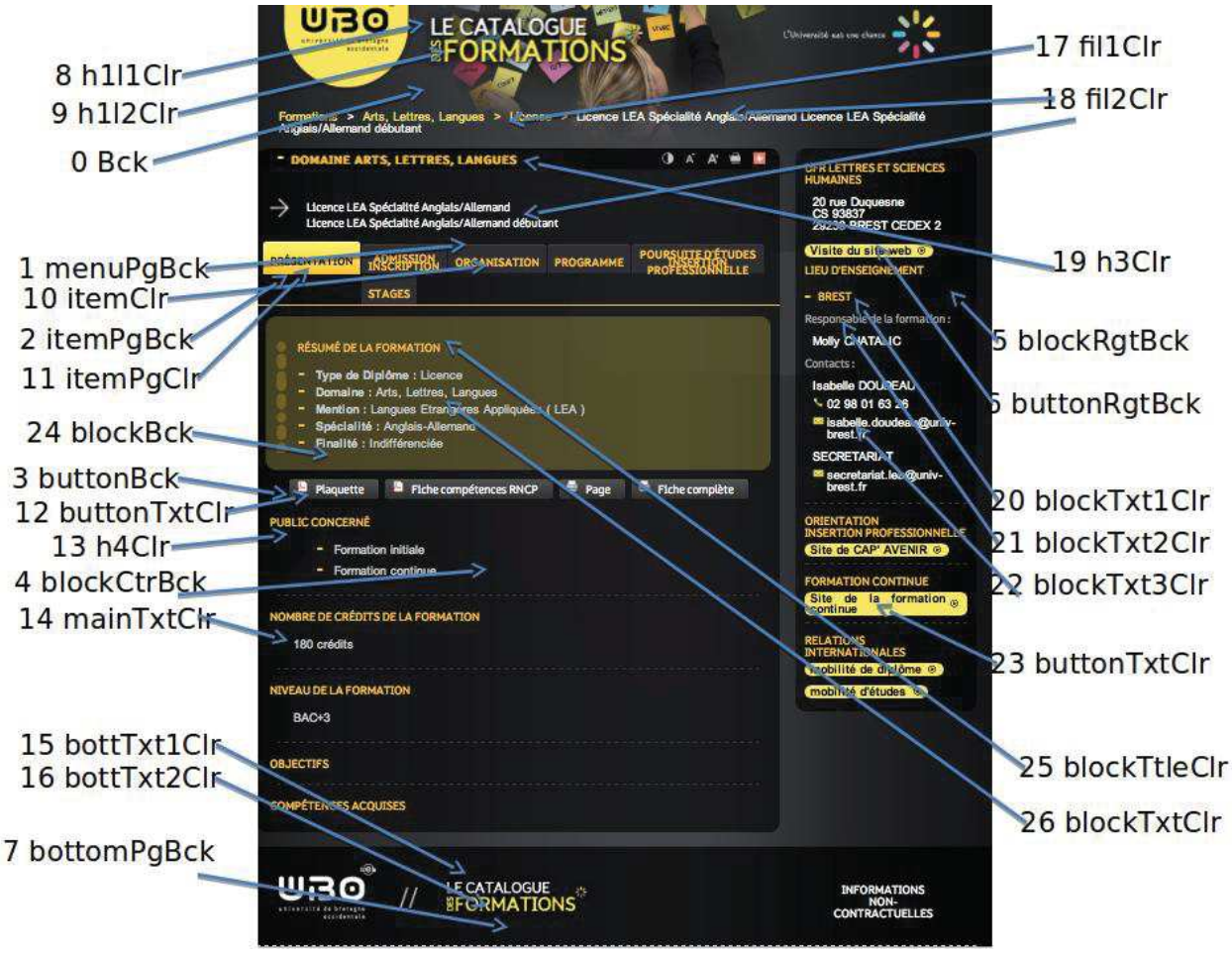


FIGURE B.9 – Page du site Web UBO Licence LEA

Variables :

0 - Bck : 38 39 42	*1 - menuPgBck : 36 36 39*
2 - itemPgBck : 249 201 56	*3 - buttonBck : 57 57 57*
4 - blockCtrBck : 20 18 20	*5 - blockRgtBck : 11 11 11*
6 - buttonRgtBck : 245 228 87	*7 - bottomPgBck : 16 16 17*
24 - blockBck : 65 62 35	*8 - h1l1Clr : 255 255 255*
9 - h1l2Clr : 246 226 80	*10 - itemClr : 246 181 48*
11 - itemPgClr : 000 000 000	*12 - buttonTxtClr : 255 255 255*
13 - h4Clr : 246 181 48	*14 - mainTxtClr : 255 255 255*
15 - bottTxt1Clr : 255 255 255	*16 - bottTxt2Clr : 246 226 80*
17 - fil1Clr : 246 226 80	*18 - fil2Clr : 255 255 255*
19 - h3Clr : 227 167 44	*20 - blockTxt1Clr : 246 181 48*
21 - blockTxt2Clr : 135 135 135	*22 - blockTxt3Clr : 255 255 255*
23 - buttonTxtClr : 000 000 000	*25 - blockTtleClr : 246 181 48*
26 - blockTxtClr : 193 193 193	

Contraintes de contraste de brillance :

$contrast(1,2) \geq x$	$contrast(0,\ 8) \geq x$	$contrast(0,\ 9) \geq x$	$contrast(0,\ 17) \geq x$
$contrast(0,\ 18) \geq x$	$contrast(1,\ 10) \geq x$	$contrast(2,\ 11) \geq x$	$contrast(24,\ 25) \geq x$
$contrast(24,\ 26) \geq x$	$contrast(3,\ 12) \geq x$	$contrast(4,\ 13) \geq x$	$contrast(4,\ 14) \geq x$
$contrast(4,\ 19) \geq x$	$contrast(7,\ 15) \geq x$	$contrast(7,\ 16) \geq x$	$contrast(5,\ 20) \geq x$
$contrast(5,\ 21) \geq x$	$contrast(5,\ 22) \geq x$	$contrast(5,\ 6) \geq x$	$contrast(6,\ 23) \geq x$
$contrast(4,\ 24) \geq x$			

Contrainte de proximité de brillance d'arrière-plans :

proxBrightness(0,1,2,3,4,5,6,7,24)

Contraintes de distance de couleurs :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$	$distOrigClr(2) \leq y$	$distOrigClr(3) \leq y$
$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$	$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$
$distOrigClr(8) \leq y$	$distOrigClr(9) \leq y$	$distOrigClr(10) \leq y$	$distOrigClr(11) \leq y$
$distOrigClr(12) \leq y$	$distOrigClr(13) \leq y$	$distOrigClr(14) \leq y$	$distOrigClr(15) \leq y$
$distOrigClr(16) \leq y$	$distOrigClr(17) \leq y$	$distOrigClr(18) \leq y$	$distOrigClr(19) \leq y$
$distOrigClr(20) \leq y$	$distOrigClr(21) \leq y$	$distOrigClr(22) \leq y$	$distOrigClr(23) \leq y$
$distOrigClr(24) \leq y$	$distOrigClr(25) \leq y$	$distOrigClr(26) \leq y$	

B.10 ReasonFrance



FIGURE B.10 – Page du site Web Reason France (forum)

Variables :

0 topMenuBck : 38 38 38	*1 topMenuSlctBck : 61 59 59*
2 generalBck : 16 16 16	*3 centralblockBck : 20 20 20*
4 topblockBck : 000 000 000	*5 infoTutoBck : 41 41 41*
6 articleBck : 26 29 31	*7 button1Bck : 62 62 62*
8 button2Bck : 40 42 45	*9 button3Bck : 80 80 80*
10 annSujBck : 27 27 27	*11 art1Bck : 30 33 37*
12 art2Bck : 25 28 30	*13 input1Bck : 255 255 255*
14 input2Bck : 179 179 179	*15 input3Bck : 40 42 45*
16 input4Bck : 80 80 80	*17 miniButt1Bck : 70 70 70*
18 miniButt2Bck : 255 255 255	*19 miniButt2SlectBck :151 151 151*
20 menuDerBck : 86 86 86	*21 buttBck : 68 68 68*
22 buttAllerBck : 80 80 80	*23 bottBck : 26 29 31*
24 topMenu1Clr : 240 140 38	*25 topMenu2Clr : 177 175 175*
26 lienRapClr : 255 255 255	*27 input1Clr : 155 155 155*
28 input2Clr : 214 214 214	*29 input3Clr : 202 202 202*
30 input4Clr : 000 000 000	*31 input4SlctClr : 196 196 196*
32 info1Clr : 255 255 255	*33 info2Clr : 240 140 38*
34 filClr : 240 140 38	*35 h1Clr : 240 140 38*
36 menuh1Clr : 240 140 38	*37 button1Clr : 29 29 29*
38 button2Clr : 202 202 202	*39 button3Clr : 000 000 000*
40 mainTxtClr : 240 140 38	*41 name1Clr : 139 28 22*
42 name2Clr : 240 140 38	*43 name3Clr : 63 147 11*
44 dateClr : 107 107 107	*45 buttClr : 30 30 30*
46 itemDerClr : 000 000 000	*47 txtDerClr : 130 130 130*
48 buttAllerClr : 000 000 000	*49 quiHeadTxtClr : 240 140 38*
50 quiTxtClr : 181 181 181	*51 permissionClr : 181 181 181*
52 links1BottClr : 240 140 38	*53 links2BottClr : 181 181 181*
54 txt1BottClr : 67 67 67	*55 txt2BottClr : 240 140 38*
56 txt2SlctBottClr : 255 255 255	*57 rechAvClr : 000 000 000*
58 rechAvSlcClr : 196 196 196	

Contraintes de contraste de brillance :

$contrast(0,24) \geq x$	$contrast(0,25) \geq x$	$contrast(1,56) \geq x$	$contrast(1,26) \geq x$
$contrast(2,26) \geq x$	$contrast(13,27) \geq x$	$contrast(14,28) \geq x$	$contrast(15,29) \geq x$
$contrast(16,30) \geq x$	$contrast(16,31) \geq x$	$contrast(4,57) \geq x$	
$contrast(4,58) \geq x$	$contrast(5,32) \geq x$	$contrast(5,33) \geq x$	$contrast(6,34) \geq x$
$contrast(3,35) \geq x$	$contrast(3,36) \geq x$	$contrast(7,37) \geq x$	$contrast(8,38) \geq x$
$contrast(9,39) \geq x$	$contrast(11,40) \geq x$	$contrast(12,40) \geq x$	$contrast(11,41) \geq x$
$contrast(12,41) \geq x$	$contrast(11,42) \geq x$	$contrast(12,42) \geq x$	$contrast(11,43) \geq x$
$contrast(12,43) \geq x$	$contrast(11,44) \geq x$	$contrast(12,44) \geq x$	$contrast(21,45) \geq x$
$contrast(20,46) \geq x$	$contrast(3,47) \geq x$	$contrast(22,48) \geq x$	$contrast(3,49) \geq x$
$contrast(3,50) \geq x$	$contrast(3,51) \geq x$	$contrast(23,52) \geq x$	$contrast(23,53) \geq x$
$contrast(3,54) \geq x$	$contrast(3,55) \geq x$		

Contrainte de proximité de brillance d'arrière-plans :

proxBrightness(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23)

Contraintes de distance de couleurs :

$distOrigClr(0) \leq y$	$distOrigClr(1) \leq y$	$distOrigClr(2) \leq y$	$distOrigClr(3) \leq y$
$distOrigClr(4) \leq y$	$distOrigClr(5) \leq y$	$distOrigClr(6) \leq y$	$distOrigClr(7) \leq y$
$distOrigClr(8) \leq y$	$distOrigClr(9) \leq y$	$distOrigClr(10) \leq y$	$distOrigClr(11) \leq y$
$distOrigClr(12) \leq y$	$distOrigClr(13) \leq y$	$distOrigClr(14) \leq y$	$distOrigClr(15) \leq y$
$distOrigClr(16) \leq y$	$distOrigClr(17) \leq y$	$distOrigClr(18) \leq y$	$distOrigClr(19) \leq y$
$distOrigClr(20) \leq y$	$distOrigClr(21) \leq y$	$distOrigClr(22) \leq y$	$distOrigClr(23) \leq y$
$distOrigClr(24) \leq y$	$distOrigClr(25) \leq y$	$distOrigClr(26) \leq y$	$distOrigClr(27) \leq y$
$distOrigClr(28) \leq y$	$distOrigClr(29) \leq y$	$distOrigClr(30) \leq y$	$distOrigClr(31) \leq y$
$distOrigClr(32) \leq y$	$distOrigClr(33) \leq y$	$distOrigClr(34) \leq y$	$distOrigClr(35) \leq y$
$distOrigClr(36) \leq y$	$distOrigClr(37) \leq y$	$distOrigClr(38) \leq y$	$distOrigClr(39) \leq y$
$distOrigClr(40) \leq y$	$distOrigClr(41) \leq y$	$distOrigClr(42) \leq y$	$distOrigClr(43) \leq y$
$distOrigClr(44) \leq y$	$distOrigClr(45) \leq y$	$distOrigClr(46) \leq y$	$distOrigClr(47) \leq y$
$distOrigClr(48) \leq y$	$distOrigClr(49) \leq y$	$distOrigClr(50) \leq y$	$distOrigClr(51) \leq y$
$distOrigClr(52) \leq y$	$distOrigClr(53) \leq y$	$distOrigClr(54) \leq y$	$distOrigClr(55) \leq y$
$distOrigClr(56) \leq y$	$distOrigClr(57) \leq y$	$distOrigClr(58) \leq y$	