Une procédure de comparaison de textes basée sur les quasi-segments : application à la traduction de rééditions

Yves Bestgen

PSOR/CECL – FNRS/UCL – B-1348 Louvain-la-Neuve –Belgique

Abstract

This study aims at proposing and testing a procedure of text comparison which can detect the presence of moves of passages in documents, a function not included in current file comparators. The detection of such kinds of modifications is important for the assistance in the translation of new editions of a book, but also for the analysis of the successive versions of a document. The procedure is based on the comparison of the quasi-segments present in the two versions. A first evaluation, based on the comparison of two editions of the same book, underlines the effectiveness of the procedure, particularly when the compared versions contain moved passages.

Résumé

L'objectif de cette recherche est de proposer et de tester une procédure de comparaison de textes capable de détecter la présence de déplacements de passages dans des documents, une fonction que les comparateurs courants n'incluent pas. Détecter ce genre de modifications est important pour l'assistance à la traduction de rééditions, mais aussi et plus généralement pour l'analyse des versions successives d'un document. La procédure s'appuie sur la comparaison des quasi-segments présents dans les deux versions. Une première évaluation, basée sur la comparaison de deux éditions d'un même livre, souligne l'efficacité de la procédure proposée, tout particulièrement lorsque les versions comparées contiennent des passages déplacés.

Mots-clés : comparaison automatique de textes, quasi-segments, révision de documents, déplacement de passages.

1. Problématique et état de la question

La présente recherche s'inscrit dans une problématique très spécifique, même si sa portée est plus générale. Elle vise à apporter une solution à un problème qui se pose lors de la mise à jour de la traduction d'un livre en raison de la parution d'une nouvelle édition en langue originale. Il est en effet nécessaire d'identifier les modifications introduites dans la nouvelle édition afin d'éviter de retraduire certains passages. Détecter manuellement ces différences est une tâche très difficile d'autant plus que les changements peuvent porter sur de grandes sections comme sur des mots ou des nombres isolés. Une solution consiste à réaliser cette comparaison de manière automatique.

Cette problématique est un cas particulier d'une situation plus générale dans laquelle on souhaite identifier les différences entre deux documents comme des programmes informatiques, des notices d'utilisation et tout autre type de documents qui doit être fréquemment mis à jour. Plus généralement, la rédaction d'une très grande variété de types de textes, y compris les articles scientifiques, implique souvent de nombreuses modifications de la part du rédacteur (Becker, 2006 ; Flower et al., 1986). Les fonctions qui permettent le suivi des modifications apportées à un document facilitent la gestion de celles-ci (Eyman et Reilly,

2006), mais le document devient difficile à lire et à manipuler dès qu'un nombre important de modifications a été introduit (Kim et Eklundh, 2001).

1.1. Les logiciels de comparaison de documents

Pour apporter une solution à ce type de problèmes, des utilitaires de comparaison de textes ont été proposés dès les années soixante-dix. Le plus connu est sans aucun doute *Diff*, initialement intégré dans le système Unix et depuis porté vers de nombreux autres systèmes (Hunt et McIlroy, 1976). Ce type d'utilitaires procède classiquement en recherchant la plus longue séquence d'éléments présents dans le même ordre dans les deux documents (Hirschberg, 1975 ; Myers, 1986). Ce qu'on entend par élément varie en fonction des objectifs de l'analyse. Lorsque la comparaison porte sur des programmes informatiques, l'objet initial de ces procédures, les éléments comparés sont le plus souvent les lignes qui composent les fichiers. Dans le cas de la comparaison de textes formatés en paragraphe, ce sont les mots ou les caractères qui se sont imposés (Fraser, 2006). De nos jours, de nombreux logiciels de comparaisons sont disponibles et des outils de ce type ont été inclus dans des suites bureautiques aussi courantes que Microsoft Word ou OpenOffice (pour une comparaison de nombreux logiciels, voir l'article *Comparison of file comparison tools* de Wikipedia).

La figure 1 présente deux versions d'une phrase telles que formulées dans la quatrième [1a] et dans la sixième [1b] édition du livre *Statistical Methods for Psychology de* Howell ainsi que les résultats de leur comparaison par *Word 2004* (Outil *Comparer des documents*) [2] et par le logiciel *DiffMerge* (V3.0) de SourceGear [3]. Les exemples [2] et [3] donnent les transformations qui doivent être appliquées à [1a] pour aboutir à [1b]. Dans ces phrases, le texte en gras (noir) est considéré comme commun aux deux versions initiales. Le texte en italique barré (bleu) est spécifique de la première version et doit donc être supprimé de celle-ci pour obtenir la seconde. Le texte en souligné (rouge) est spécifique de la seconde version et doit donc être inséré pour aboutir à cette seconde version.

[1a] *These p values can be calculated using commonly available programs or they can be taken from standard computer printouts.*

[1b] These p values can be taken from standard computer printouts, or they can be calculated using commonly available programs.

[2] **These p values can be** taken from standard computer printouts, or they can be **calculated using commonly available programs** ~~or they can be taken from standard computer printouts~~**.**

[3] **These p values can be** taken from standard ~~calculated using~~ **computer** printout~~monly available programs~~**s, or they can be** calculated using commonly available program~~ taken from standard computer printouts~~**.**

Figure 1 : Application de deux comparateurs à une phrase : versions originales [1a] et [1b], comparaison selon Word [2] et selon DiffMerge [3].

On constate que pour ce très court exemple, le comparateur de *Word* est légèrement plus performant que *DiffMerge* puisqu'il identifie à raison 10 mots communs contre 9.

1.2. Problèmes pour la traduction de rééditions

Par rapport au but spécifique que nous visons, l'assistance à la traduction de rééditions, ces logiciels et outils présentent trois limitations principales auxquelles cette recherche tente de répondre. Tout d'abord, certains logiciels établissent des correspondances entre des portions de mot comme l'indique l'exemple [3]. Établir ce genre de correspondances peut présenter un intérêt pour certaines applications (comme réduire l'espace nécessaire pour le stockage de multiples versions d'un même document), mais est peu pertinent lors de la comparaison de textes en général, et dans le contexte d'une traduction en particulier. Pour ces tâches, c'est la détection de mots communs qui s'impose comme à l'exemple [2]. Il faut néanmoins noter qu'identifier des changements portant, par exemple, spécifiquement sur la désinence d'un verbe peut avoir un certain intérêt.

Une deuxième limitation, liée à la précédente, résulte de ce que ces logiciels ont été conçus pour identifier un maximum d'éléments communs, même si ceux-ci correspondent à des mots isolés au milieu de nombreux mots non communs. La figure 2, résultant de l'analyse par DiffMerge des quatrième et sixième éditions du livre de Howell, met en évidence les difficultés que ce mode de fonctionnement peut provoquer pour la révision d'un texte : [4a] donne la version à laquelle il faut aboutir et 4b montre quels mots peuvent être récupérés de l'ancienne version pour y parvenir.

[4a] ... As a second example, the probability of 6 correct choices out of 10 trials is the probability of any one such outcome.

[4b] ... ~~We are concerned with~~ the probability of ~~obtaining 9~~ correct choices ~~(and 1 incorrect choice)~~ out of 10 trials~~. One outcome that satisfies our criterion would be nine corrects followed by one incorrect, and~~ the probability of ~~this particular~~ outcome ~~is the joint probability~~.

Figure 2 : Résultats produits par DiffMerge lors de la comparaison de deux documents.

La troisième limitation de ces programmes est qu'ils n'admettent que deux opérations de modification d'un texte : la suppression de certains éléments et, son pendant, l'ajout d'éléments. Tout particulièrement, ils ne signalent pas spécifiquement les passages qui ont été déplacés comme dans la figure 1 où les segments « *using commonly available programs* » et « *or they can be* » ont été déplacés lors de la révision du texte. La figure 3 présente une analyse qui met en évidence ce déplacement. Les passages en gras et italique ou en gras et souligné sont présents dans les deux versions, mais ont été déplacés de telle manière qu'ils apparaissent dans une version avant et dans l'autre après un même segment commun (*taken from standard computer printouts*).

[5] These p values can be ~~calculated using commonly available programs. or they can be~~ taken from standard computer printouts, or they can be calculated using commonly available programs.

Figure 3 : Mise en évidence des passages déplacés dans les phrases [1a] et [1b] (figure 1).

L'absence de l'opération de déplacement s'explique par la complexité et le coût en temps calcul qu'implique la recherche de ce genre de modifications (Cobéna et al., 2004 ; Monostori et al., 2001). Il s'agit pourtant d'un type de modifications fréquent lors de la rédaction ou lors

de la révision d'un document (Eyman et Reilly, 2006 ; Horning, 2002). Dans le cas de la traduction d'éditions successives, ces opérations, non détectées, entraînent un surcroît de travail de la part du traducteur. Par exemple, la révision pour la sixième édition du chapitre 11 du livre de Howell inclut 9 déplacements de passages pour un total de plus de 1 700 mots, soit plus de 9% du texte original.

L'objectif de ce travail est de proposer une procédure de comparaison de textes qui remédie à ces trois limitations. La section suivante décrit la solution proposée et son implémentation. La troisième section présente une première évaluation de l'efficacité de cette procédure dans la détection des passages identiques ou différents dans deux éditions d'un ouvrage.

2. Procédure proposée et implémentation

L'analyse qui vient d'être faite indique que les procédures de comparaison actuellement disponibles présentent plusieurs lacunes pour l'analyse des modifications apportées lors de rééditions d'un ouvrage. La procédure proposée ici vise à remédier à ces limitations.

2.1. Principes généraux

Le problème majeur que rencontrent les procédures de comparaison existantes porte sur l'impossibilité d'identifier adéquatement les passages d'un document qui ont été déplacés lors de sa révision. Pour le résoudre, il est indispensable que la procédure soit capable d'identifier des segments communs quelque soit l'ordre dans lequel ces segments apparaissent dans les deux versions du document. Cette condition est rencontrée par l'identification automatique des séquences de mots, aussi appelées *N-grams,* fréquemment employée en traitement automatique du langage, par exemple dans la reconnaissance automatique de la parole (Manning et Schütze, 2001) et dans l'identification des expressions phraséologiques (Hernandez et Grau, 2003 ; Piérard et Bestgen, 2006). Tout particulièrement, cette technique a été employée pour détecter des cas de plagiat, un domaine proche de la comparaison des versions d'un même document, mais, plus complexe, car il impose de rechercher des similarités entre un document et de très nombreux autres. Lyon et al. (2001) et van Halteren (2003) ont proposé de rechercher des ressemblances entre des documents sur la base des trigrammes qui les composent. Ils en dérivent une sorte « d'empreinte digitale » du document qu'ils comparent à celles d'autres documents. Une faiblesse de cette approche est qu'elle ne permet pas d'identifier finement les passages des documents qui sont similaires. Monostori et al. (2001b) ont proposé d'identifier les cas de plagiat en recherchant les plus longues séquences de caractères identiques dans deux ou plusieurs documents. Monostori et al. (2001a) décrivent l'application de cette technique à la comparaison d'éditions successives d'ouvrages. Les limitations principales de leur approche sont que seules les séquences identiques sont prises en compte et que l'unité d'analyse est le caractère et non le mot.

L'identification automatique des séquences de mots est également employée très fréquemment en statistique textuelle (Lebart et Salem, 1994). Il faut toutefois noter que l'approche proposée ici est un cas très particulier de l'analyse des segments répétés. On recherche, en effet, des segments qui sont *répétés* dans le sens qu'ils apparaissent dans les deux versions d'un même document, et qui sont, simultanément, autant que possible *uniques* dans chaque version afin d'éviter des « collisions », c'est-à-dire des situations dans lesquelles un segment apparaît plusieurs fois dans une version et une ou plusieurs fois dans l'autre. Cette double exigence impose que la longueur des segments soit suffisamment courte pour qu'on puisse identifier des séquences communes aux deux versions et suffisamment longue pour éviter des répétitions à l'intérieur des versions. Cette exigence peut être rencontrée par un

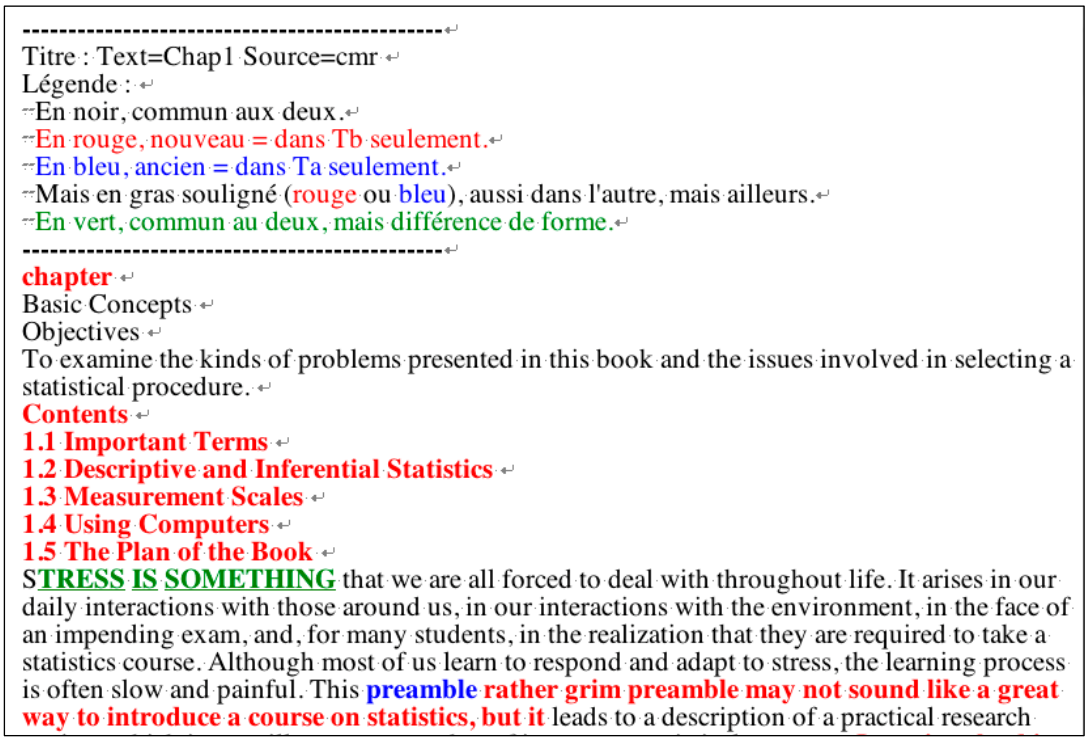
processus itératif qui effectue une première analyse avec des segments longs, identifie autant de mots communs que possible et les rend uniques afin qu'ils puissent servir de point de repère pour les analyses ultérieures qui rechercheront des segments plus courts.

Une deuxième difficulté que la procédure doit prendre en compte est qu'on souhaite identifier comme communs des segments même si l'un ou l'autre mot ont été modifiés. En effet, il est important qu'une phrase entière ne soit pas identifiée comme neuve alors que seuls un ou deux mots ont été modifiés. Cette contrainte impose d'identifier des « quasi-segments » (pour reprendre un terme proposé par Becue, voir Lebart et Salem, 1994, p. 124), des séquences de mots dont un nombre suffisant, mais pas nécessairement tous, sont identiques dans les deux versions du document. Pour ce faire, des jokers sont introduits dans les segments recherchés. Par exemple, le segment de longueur 5 « a b c d e » peut, selon ce principe, être considéré comme identique aux trois segments suivants : « a **f** c d e », « a b **g** d e » et « a b c **h** e ». Les segments « **i** b c d e » et « a b c d **j** » peuvent également être considérés comme identiques au segment de départ, mais il semble plus logique de les considérer comme des segments de longueur 4 et de les traiter lors d'une itération ultérieure.

Un des avantages d'une procédure basée sur les quasi-segments est qu'elle rend peu probable l'identification comme commun d'un mot entouré de mots tous considérés comme différents. Une telle situation n'est cependant pas totalement impossible. Ce problème, également présent dans d'autres procédures de comparaison, peut être résolu en calculant pour chaque mot commun la densité de mots communs au voisinage de celui-ci et en éliminant les mots pour lesquels cette densité est trop faible.

2.2. Procédure QS : implémentation

La procédure de comparaison de textes, appelée *QS* en raison de son recours aux quasi-segments, met en pratique les principes décrits au point précédent en fixant les paramètres suivants. L'analyse est effectuée en deux étapes, la première est basée sur des quasi-segments de longueur 9 et la seconde sur des quasi-segments de longueur 7. Ces quasi-segments sont centrés sur le mot pour lequel on cherche à déterminer s'il est identique ou non dans les deux versions. Chaque quasi-segment accepte deux mots différents, l'un à gauche et l'autre à droite du mot central. L'ensemble des quasi-segments est extrait de chaque version et comparé afin d'identifier des segments identiques. Pour le critère de densité, un mot n'est considéré comme commun que si, parmi les 40 mots qui l'entourent (20 à gauche et 20 à droite), il y en a au moins 20% qui sont aussi considérés comme commun.



Titre : Text=Chap1 Source=cmr
Légende :
En noir, commun aux deux.
En rouge, nouveau = dans Tb seulement.
En bleu, ancien = dans Ta seulement.
Mais en gras souligné (rouge ou bleu), aussi dans l'autre, mais ailleurs.
En vert, commun au deux, mais différence de forme.
---
chapter
Basic Concepts
Objectives
To examine the kinds of problems presented in this book and the issues involved in selecting a statistical procedure.
Contents
1.1 Important Terms
1.2 Descriptive and Inferential Statistics
1.3 Measurement Scales
1.4 Using Computers
1.5 The Plan of the Book
STRESS IS SOMETHING that we are all forced to deal with throughout life. It arises in our daily interactions with those around us, in our interactions with the environment, in the face of an impending exam, and, for many students, in the realization that they are required to take a statistics course. Although most of us learn to respond and adapt to stress, the learning process is often slow and painful. This preamble rather grim preamble may not sound like a great way to introduce a course on statistics, but it leads to a description of a practical research

Figure 4 : Sorties formatées de la procédure QS.

La procédure a été écrite en SAS (Statistical Analysis System, V6.12), section Base et Statistics, sous la forme de macros appelées par trois programmes. Le premier transforme les textes en fichiers de données SAS. Le second effectue la comparaison et le troisième génère le fichier de résultat au format HTML avec un codage par couleur et style des différents types de différences identifiées. La figure 4 présente un exemple de sortie de la procédure.

L'utilisation d'un logiciel de statistique pour implémenter cette procédure en réduit l'efficiente en termes de temps calcul. Par contre, cela facile la mise au point et donne accès à des procédures statistiques qui permettent d'effectuer des contrôles lors du développement de la procédure, mais aussi lors de son utilisation. Il est à noter que, vu la lourdeur du travail d'édition et de traduction qui doit être effectué sur les sorties de la procédure, le temps calcul est un paramètre nettement moins important que le nombre de mots communs identifiés.

3. Évaluation

L'objectif de l'évaluation est de comparer l'efficacité de la procédure proposée à celle d'autres procédures. La mesure de l'efficacité est basée sur le nombre de mots détectés à raison comme commun aux deux versions d'un même document. Un mot est considéré comme commun lorsque tous les caractères qui le composent sont considérés comme communs. Cet indice a été préféré à celui basé sur le nombre de caractères communs parce que, comme expliqué ci-dessus, l'unité pertinente pour les traitements ultérieurs est le mot. Comme je n'ai pas connaissance d'un matériel de test développé spécifiquement pour ce type de comparaison et comme créer ce genre de matériel aurait posé des problèmes de représentativité (quelle proportion de modifications introduire, combien de déplacements, etc.), j'ai employé un matériel authentique : la quatrième (1997) et la sixième édition (2007) du livre *Statistical Methods for Psychology de* Howell.

3.1. Sélection des procédures comparées

Dans un premier temps, une étude pilote a été menée afin de sélectionner des logiciels auxquels comparer la procédure proposée. Sur la base des indications données sur la page *Comparaison of file comparison tools* de Wikipedia et de celles obtenues grâce à une recherche sur internet, six programmes ont été installés dans leur version Windows ou Mac OS X. Le critère principal de sélection était qu'il soit librement utilisable pour des objectifs de recherche ou qu'une version non bridée puisse être employée pendant une période d'essai. Il s'agit des logiciels : *Differences Examiner (V2.1)* de *AlphaOmega Software*, *DiffMerge (V3.0)* de *SourceGear*, *ExamDiff Pro (V3.5)* de *prestoSoft*, *Merge (V6.5)* de *Araxis*, *DiffDoc (V3.70)* de *Softinterface* et *Compare&Merge (V2.3a)* de *TGRMN Software*. À ceux-ci ont été ajoutés le comparateur inclus dans les outils de *Microsoft Word 2004* pour Mac OS X et la procédure proposée.

Le matériel pour cette étude pilote était constitué par la préface de la quatrième et de la sixième édition du livre de Howell. La comparaison a porté sur l'efficacité des procédures et, surtout, sur la possibilité d'extraire de manière largement automatique des sorties des logiciels un indice d'efficacité fiable. En effet, ces sorties ont été systématiquement formatées pour permettre à l'utilisateur de détecter les similarités et différences entre deux textes et de construire sur cette base une version optimale. Le calcul d'un indice global de ressemblance entre les textes n'est donc pas prévu, ou lorsqu'il l'est, il est inutilisable pour la comparaison comme, par exemple lorsque cet indice est formulé en termes de nombre de lignes différentes.

Deux programmes seulement ont été identifiés comme permettant une extraction relativement simple du nombre de mots communs identifiés, *simple* signifiant ici qu'il est possible de calculer ce nombre en effectuant des suppressions sélectives de caractères sur la base de leur formatage : couleur, style, etc. Il s'agit de l'outil de comparaison de *Microsoft Word* et de *DiffMerge* de *SourceGear*. Ce deuxième programme étant également parmi les plus efficaces de tous ceux qui ont été comparés, l'analyse complète a été limitée à ces deux logiciels.

3.2. Comparaison des trois procédures

Cette comparaison a porté sur les quatrième et sixième éditions des 12 premiers chapitres du livre de Howell. La longueur moyenne de ces chapitres est de 13 756 mots dans la quatrième édition (écart-type = 5 607 ; min = 6 493 ; max = 22 149) et de 16 165 mots dans la sixième (écart-type = 6 683 ; min = 6 456 ; max = 25 598). Avant d'analyser l'efficacité des trois procédures, il faut souligner que l'outil de comparaison de *Word* est le plus rapide (11 secondes), suivi par *DiffMerge* (52 secondes) et puis par la procédure *QS* (165 secondes)[1].

La figure 5 permet de visualiser les différences d'efficacité entre *QS* et les deux autres procédures. Le graphique de gauche donne pour chaque chapitre le gain d'efficacité de *QS* par rapport à *Word* (WO). L'indice est obtenu en soustrayant le nombre de mots identifiés comme communs par *Word* du nombre de mots identifiés comme communs par *QS*, le tout divisé par le nombre de mots de la plus courte des deux versions du chapitre. Plus cet indice est élevé, plus *QS* est efficace par comparaison à *Word*. Le graphique de droite donne les mêmes informations pour la comparaison entre *QS* et *DiffMerge* (DM). La comparaison entre *Word* et

[1] Valeurs pour la comparaison de deux textes de plus ou moins 22 000 mots sur un Power Mac G5 (2.5 GHz), environnement *OS X* pour *Word* et *DiffMerge*, *Classique* pour *QS*.

DiffMerge n'est pas donnée explicitement parce qu'elle n'est pas l'objet de la présente étude et qu'elle peut être inférée en prenant *QS* comme point de repère.

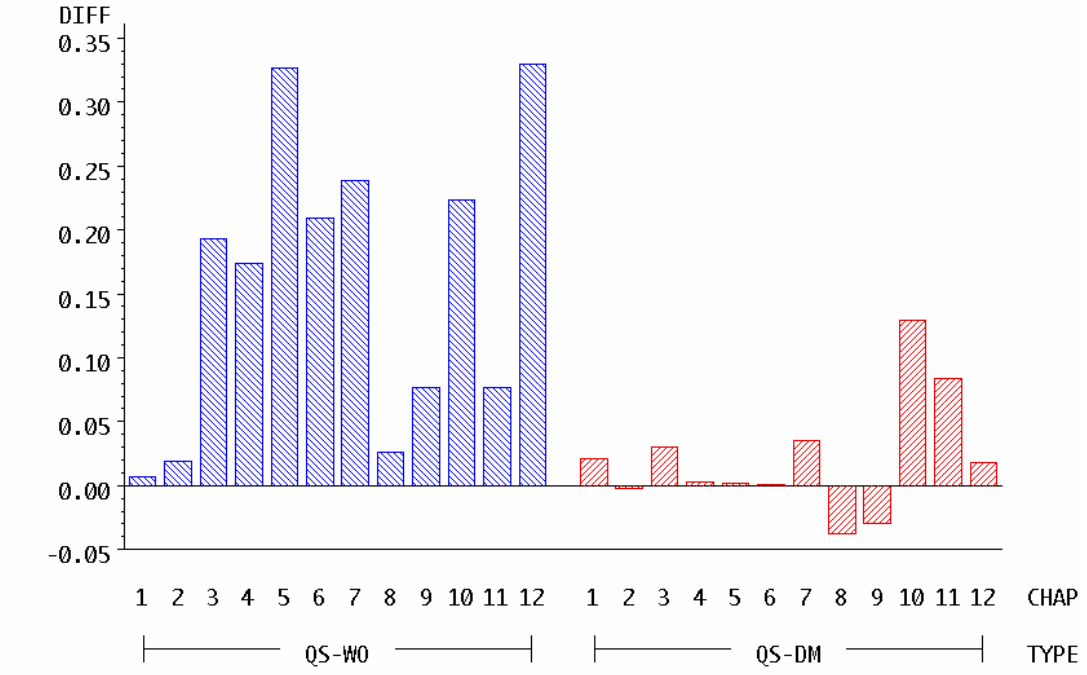


Figure 5 : Différences d'efficacité (en proportion) entre QS et les deux autres procédures pour les 12 chapitres.

Manifestement, l'outil de comparaison de *Word* est le moins performant. Lorsqu'il y a peu de modifications, il s'en sort bien, mais dès que celles-ci sont nombreuses, il n'arrive plus à établir des correspondances entre les deux versions. Les performances de *QS* et de *DiffMerge* sont identiques pour 4 des 12 chapitres ; celles de *DiffMerge* sont supérieures à celles de *QS* pour 2 chapitres et inférieures pour les 6 chapitres restants. Tout particulièrement, on observe que *QS* surpasse nettement *DiffMerge* dans l'analyse des chapitres 10 et 11. Il s'agit des chapitres dans lesquels *QS* a identifié la plus grande proportion de passages déplacés d'une version à l'autre. Cette analyse est confirmée par l'existence d'une très forte corrélation positive entre la différence d'efficacité pour ces deux procédures et le nombre de mots déplacés (N=12, r=0.84, p<0.001). C'est donc bien à ce niveau que *QS* apporte une amélioration. Par contre, lorsque les déplacements sont rares, *DiffMerge* est légèrement plus efficace que *QS*.

4. Discussion et conclusion

L'objectif spécifique de cette recherche était de proposer et de tester une procédure de comparaison de textes capable de détecter la présence de déplacements de passages dans des documents. Identifier ce genre de modifications est important dans le cas de la traduction de rééditions d'ouvrages. Plus généralement, déplacer des constituants syntaxiques, des phrases et des paragraphes est une opération fréquente lors de la rédaction et de la révision d'un texte (Horning, 2002). La procédure proposée tente d'atteindre cet objectif en s'appuyant sur la comparaison des quasi-segments présents dans les deux versions d'un document.

L'évaluation confirme l'efficacité de la procédure tout en soulignant ces limites. *QS* surpasse nettement la procédure *DiffMerge* lorsque des déplacements multiples sont présents. Par contre, en l'absence de ceux-ci, l'efficacité de *DiffMerge* est au moins égale et même

supérieure dans certains cas. Il s'agit donc là d'une première limitation de ce travail. Il doit être possible d'améliorer *QS* afin qu'il soit, en l'absence de tout déplacement, au moins aussi efficace que les autres procédures de comparaison. Une autre limitation de *QS* est son manque d'efficience en termes de temps calcul. Une part de celui-ci est évidemment liée au langage de programmation employé. Parallèlement, un gain appréciable de vitesse devrait pouvoir être obtenu en modifiant la procédure pour qu'elle fonctionne en deux temps : d'abord, rechercher, conformément à l'algorithme à la base de *Diff*, la plus longue séquence d'éléments présents dans le même ordre dans les deux documents et ensuite s'affranchir de l'ordre original et rechercher les passages déplacés.

Une autre limitation de cette recherche est que *QS* n'a été comparé qu'à deux autres procédures dans l'expérience principale. Il faut cependant noter que la procédure *DiffMerge* est sortie parmi les meilleures de l'étude pilote qui comparait 6 logiciels. Enfin, les résultats présentés n'ont été obtenus que par l'analyse d'un seul ouvrage. D'autres comparaisons permettraient de confirmer le gain d'efficacité apporté par la prise en compte des passages déplacés.

Ces limitations sont autant de pistes pour de nouvelles recherches. Il serait également intéressant d'envisager d'autres approches que celles des quasi-segments pour rechercher les passages déplacés. Une analyse des procédures en cours de développement dans le champ très dynamique de l'étude du plagiat pourrait être très fructueuse. Il s'agit en effet d'une situation très comparable à celle étudiée ici puisque les passages réutilisés peuvent se situer n'importe où dans le texte et ont fréquemment été altérés de diverses manières.

Références

Becker A. (2006). A review of writing model research based on cognitive processes. In Horning A. and Becker A. editors, *Revision: History, Theory, and Practice* (pp. 25-49). Parlor.

Eyman D., and Reilly C. (2006). Revising with word processing/technology/document design. In Horning A. and Becker A. editors, *Revision: History, Theory, and Practice* (pp. 102-116). Parlor.

Flower L., Hayes J. R., Carey L., Schriver K., and Stratman J. (1986). Detection, diagnosis, and the strategies of revision. *College Composition and Communication*, 37: 16-55.

Fraser N. (2006). Diff Strategies. [http://neil.fraser.name/writing/diff/ - Version d'avril 2006].

Hernandez N. and Grau B. (2003). Automatic extraction of meta-descriptors for text description. In *Proceedings of the International Conference on Recent Advances In Natural Language Processing (RANLP)*. Borovets, Bulgaria.

Hirschberg D. S. (1975). A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*, 18: 341–343.

Horning A. S. (2002). *Revision Revisited*. Hampton Press.

Howell D. C. (1997/2007). *Statistical Methods for Psychology*. Thomson.

Hunt J. W. and McIlroy M. D. (1976). An algorithm for differential file comparison. *Computing Science Technical Report*, Bell Laboratorie.

Kim H. C. and Eklundh K. S. (2001). Reviewing practices in collaborative writing. *Computer Supported Cooperative Work*, 10: 247–259.

Lebart, L. et Salem, A. (1994). *Statistique textuelle*. Dunod.

Lyon, C., Malcolm, J. and Dickerson, B. (2001). Detecting short passges of similar text in large document selections. In *Conference on Empirical Methods in Natural Language Processing (EMNLP2001)*, pages 118-125.

Manning C. and Schütze H. (1999). *Foundations of Statistical Natural Language Processing*. MIT.

Monostori K. Zaslavsky A. and Bia A. (2001a). Using the MatchDetectReveal system for comparative analysis of texts. In *Proceedings of the Sixth Australian Document Computing Symposium (ADCS 2001)*, pages 51-58.

Monostori K., Zaslavsky A. and Schmidt H. (2001b). Parallel and distributed document Overlap detection on the Web. In Sørevik T., Manne F., Moe R. and Gebremedhin, editors, *Proceedings of the 5th InternationalWorkshop on Applied Parallel Computing (PARA 2000)*, pages 206-214.

Myers E. (1986). An O(ND) Difference algorithm and its variations. *Algorithmica*, 1: 251–266.

Piérard S. et Bestgen Y. (2006). A la pêche aux marqueurs linguistiques de la structure du discours. In *Actes des 8es Journées internationales d'Analyse statistique des Données Textuelles (JADT08),* pages 749-757.

van Halteren H. (2003). Detection of plagiarism in student essays. In Decadt B., Hoste V., and De Pauw G., editors, *Proceedings of Computational Linguistics in the Netherlands*.

Wikipedia. *Comparison of file comparison tools*. [http://en.wikipedia.org/wiki/Comparison_of_file_ comparison_tools - version du 19.09.2007].

Note de l'auteur :

Yves Bestgen est chercheur qualifié du Fonds national de la recherche scientifique (FNRS). Cette recherche est financée par une « Action de Recherche concertée » du Gouvernement de la Communauté française de Belgique.