Ilhan ACHERCHOUR / Hakim AOUDIA
Semestre 4 - 2ème année de Licence

03/04/2022

Algorithmique des arbres

Rapport de l'étape 2

Sommaire :

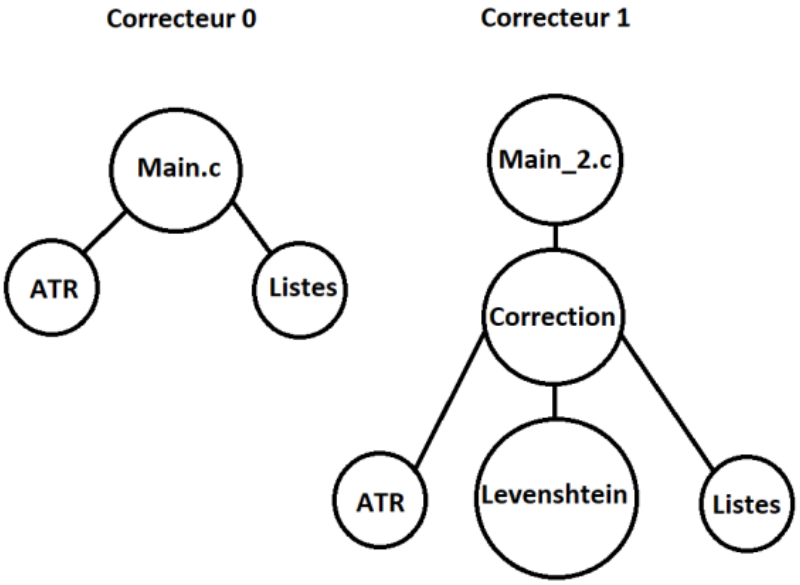
Introduction

Après avoir réalisé un programme permettant de détecter les mots mal orthographiés (selon un dictionnaire) nous devions cette fois donner une possible correction de ses mots à l'aide de la distance de Levenshtein. Nous allons parcourir chaque mot du dictionnaire et calculer la distance de Levenshtein avec le mot à corriger pour trouver une correction.

Compilation

Pour cette deuxième étape nous avons modifier le makefile, on compilera à l'aide de cette manière :
makefile Projet
./correcteur_1 texte_a_corriger.txt dictionnaire.dico

On peut schématiser la relation des fichiers entres eux de cette manières :



Difficulté

Fonction Levenshtein

Pour réaliser cette fonction nous nous sommes inspirés de la méthode itérative. Nous aurions pu faire la version récursive mais d'après le sujet elle n'était pas très optimal car elle parcourait plusieurs fois la même distance. Cette fonction ne nous a pas posé de problème.

Algorithme principal

Cet algorithme nous a posé plus de problèmes que la fonction de levenshtein. On voulait au départ parcourir tous les mots de l'ATR dictionnaire que nous avons créé à l'étape une cependant lorsqu'on rajouter les possibles correction des mots dans une liste chaînée cela ne fonctionne pas. Cela rajoute seulement le dernier mots du dictionnaire pour une raison que l'on ignore encore. Nous avons vérifié si le problème venait de insere_tete() mais cette fonction marche.
On a alors changé de façon de parcourir les mots du dictionnaire. Et on parcourt alors tous les mots du dictionnaire à l'aide de fscanf(). Avec cette méthode notre algorithme fonctionne et nous propose les bons comme correction des mots mal orthographiés.

Nous avons laissé la première méthode parcourant un ATR pour montrer que nous avons tout même essayer avec cette technique. Cependant on utilise le parcours de fichier pour notre algorithme.

Documentation utilisateur

Pour l'utilisateur il suffit de lancer le programme avec en premier paramètre le texte à corriger et en deuxième paramètre le dictionnaire. Cela va ensuite afficher la liste des mots à corriger avec une ou plusieurs propositions.

Exemple :
./correcteur_1 a_corriger_0.txt dico_3.dico

Conclusion

On a bien aimé réaliser cette deuxième étape car elle permet de comprendre comment les correcteurs orthographiques que nous utilisons tout les jours fonctionnent. De plus, cette deuxième étape ne nous a pas posé autant de difficulté que le premier rendu.