PROJET DE PROGRAMMATION C

Rapport d'expériences

Sujet:
VAMOS A LA PLAGIAT

Elaboré par:
Sabana SURESH
Oscar PERIANAYAGASSAMY
Groupe de TD6

Encadrants:
Emmanuel Lazard
Florian Sikora

Université Paris Dauphine PSL – L2 MIDO 2022/2023

Dans ce rapport d'expériences nous allons vous présenter des expériences qui nous ont permis de tester notre programme et de corriger si nécessaire certains bugs.

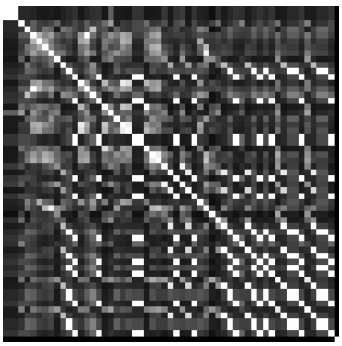
Quand l'algorithme hongrois est utilisé, c'est précisé. Dans le cas contraire c'est que l'algorithme de couplage est l'algorithme glouton proposé dans le sujet.

De plus, la validité des algorithmes présents dans les fichiers n'est pas forcément assurée. Le but étant d'étudier le comportement de notre programme vis-à-vis de modifications effectuées globalement (déplacement de portions de code ou inclusion par exemple), cela n'était pas nécessaire de travailler avec du code logique.

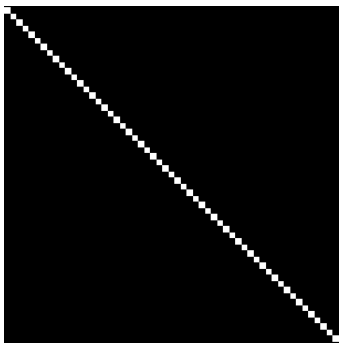
Expérience 1 : diverses manipulations autour d'un programme courant d'une cinquantaine de lignes

Pour cette première expérience, nous nous sommes fixés une idée de sujet et avons essayé de l'implémenter. Il s'agit d'un programme qui lit dans un fichier une suite de caractères, qui les stocke dans un tableau et qui trie ce tableau. Nous avons décidé d'utiliser le $Merge\ Sort$ pour trier. La version de départ est $e1\_o.c$. Elle est déclinée en plusieurs « sous-versions » que l'on détaille dès maintenant.

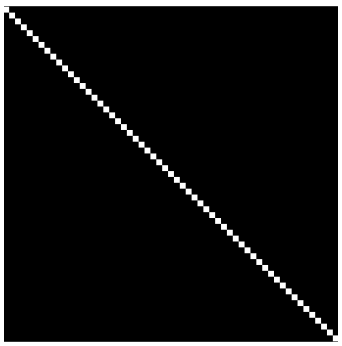
Avant tout, nous avons comparé le fichier avec lui-même, obtenu une distance de 0 et les images suivantes :



Représentation des distances de Dice



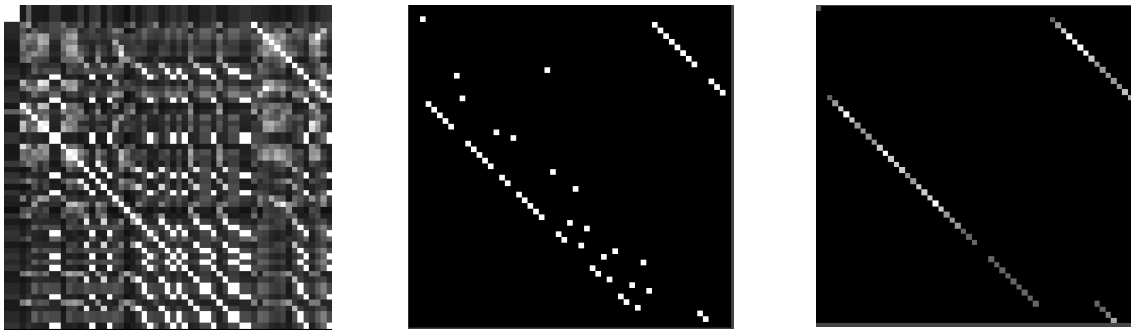
Représentation des segments couplés



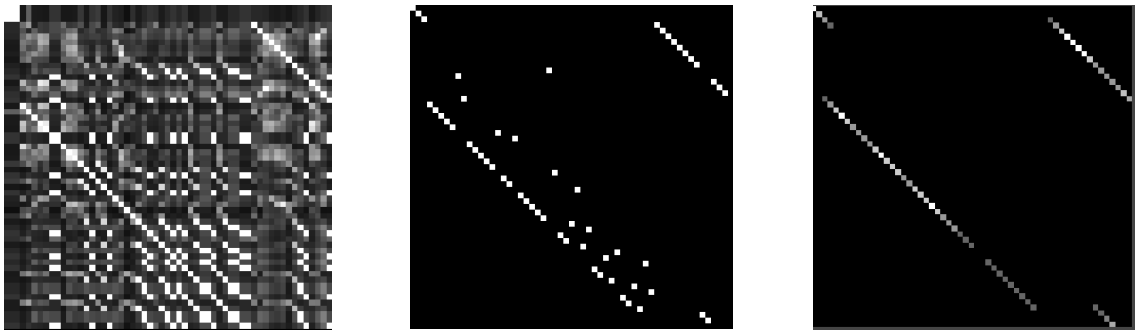
Représentation des segments couplés après le post-filtrage

Dans ce cas, c'est bien ce que l'on doit avoir. La diagonale dans les images de couplage et de post-filtrage nous montre bien la correspondance parfaite des lignes une à une entre les deux fichiers.

La seconde comparaison est faite avec le fichier $e1\_depla.c$ qui correspond au fichier de base dans lequel l'ordre du $main$ et les fonctions de tri a été inversé. Une distance de 0.38 est calculée et les images sont :

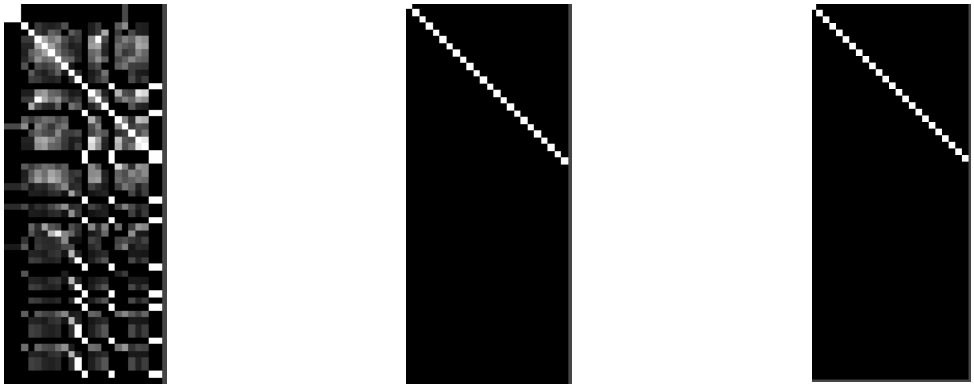


Notons qu'avec l'algorithme hongrois, la distance obtenue est de 0.33 et sont obtenues :



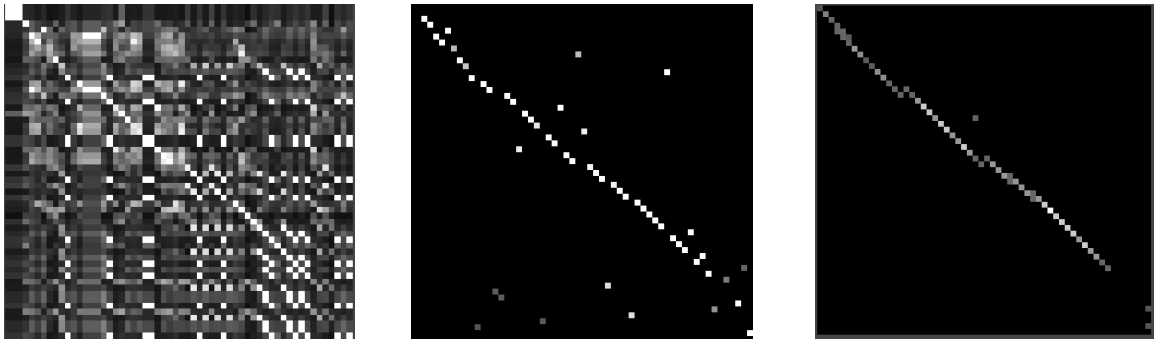
Ces résultats sont assez logiques car en déplaçant le bloc, les lignes égales ne coïncident plus. Autrement dit, le couplage ne génère plus une belle diagonale mais on peut observer deux droites symétriques, témoignant de lignes consécutives plagiées.

Ensuite, nous avons testé l'inclusion. Le fichier $e\_inclus.c$ est le même que $e\_o.c$ à la différence que la fonction $Merge()$ a été retirée. Assez logiquement nous obtenons une distance de 0.00.



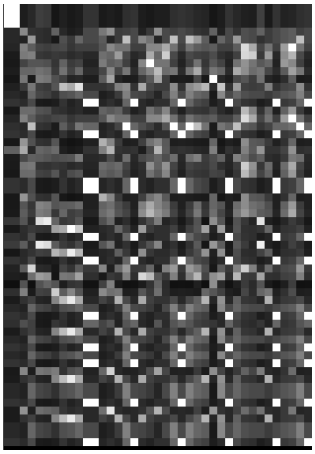
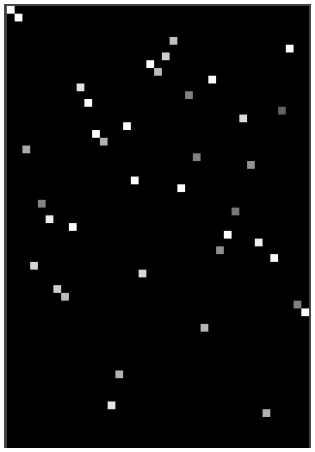
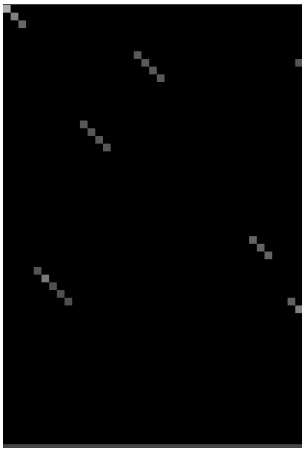
Chaque segment du fichier inclus a en effet un autre segment avec lequel il partage une distance de Dice nulle.

Testons désormais avec un plagiat camouflé. Le code d'origine a été copié et légèrement modifié dans le fichier $e1\_petite\_modif.c$. « Légèrement modifié » correspond principalement à des changements de nom de variables, déplacements de code, autre implémentation de la fonction $Merge()$. La distance est de 0.44.



L'expérience 1 s'achève avec la comparaison d'un fichier codé par une personne neutre vis-à-vis du sujet. Grâce à ChatGPT, une intelligence artificielle conversationnelle développée par OpenAI, il nous a été possible de générer un autre code source pour le problème initial. Il est stocké dans $e1\_ai.c$. La question qui lui a été posée a été : « Can you generate a c program that reads a file fulfilled with chars, stores these chars in an array and sorts the array ? ».

Ce test a pour but de voir si le programme pouvait détecter injustement des similitudes entre deux fichiers écrits par deux codeurs différents et indépendants. Comme prévu, les deux fichiers sont séparés d'une distance s'élevant à 0.78.

La similitude est assez ponctuelle et répartie au sein du fichier. Cela ne permettrait pas de conclure de la présence de plagiat.

Expérience 2 : test de robustesse et d'inversion

Nous avons trouvé un projet assez long de simulation de match de tennis en ligne (en libre téléchargement). De plus, le père d'Oscar nous a gentiment transféré un petit projet de C qu'il a réalisé lors de ses études. En additionnant ces deux données et les programmes de l'expérience précédente, il nous a été possible de produire un fichier de 1845 lignes (en comptant les lignes vides et les commentaires). Nous l'avons un petit peu mélangé dans tous les sens pour obtenir un fichier à comparer. Il s'agit de $testg.c$ et $testg2.c$. Leur distance est de 0.24 (voir fin de document pour les images).

Ces fichiers n'ont en soit aucun sens, mais c'était intéressant de tester sur un assez gros volume de données en entrée.

Le test d'inversion consiste à vérifier que $./main\ testg.c\ testg2.c$ et $./main\ testg2.c\ testg.c$ renvoie la même distance. Pour l'algorithme glouton, il y a une différence de 0.01 entre les deux résultats. Cela peut s'interpréter par le fait que le couplage renvoyé n'est pas forcément minimal. L'algorithme hongrois, quant à lui, retourne deux fois la valeur 0.23.

Conclusion

Nous pouvons donc conclure que cette série de tests est plutôt concluante. Nous avons mis à l'épreuve notre programme sur une liste non-exhaustive de cas. Cependant, celle-ci nous paraît assez représentative de situations qui pourraient

réellement se produire. Par exemple, dans le cas d'un TP à rendre, le plagiat est fréquent et réalisé de manière assez élémentaire.

D'autres expériences ont également été effectuées pour tester notre programme, nous aurons le plaisir de vous en présenter quelques-unes lors de notre soutenance le 10/02/2023.

Annexe