

HLIN608 - Algorithmique du texte - TD Assemblage

Denis BEAUGET, Antoine AFFLATET et Jérémie ROUX (L3 Groupe C)

2019 - 2020

Exercice 1

Soit $F = \{F_1, F_2, F_3, F_4, F_5\}$ tel que :

$$F_1 = ACCTGAG$$

$$F_2 = TGCATTGC$$

$$F_3 = GCAGACC$$

$$F_4 = AGCAAT$$

$$F_5 = CAATG$$

Appliquons la méthode gloutonne pour extraire la plus petite chaîne F comprenant tout ces sous-mots. Pour la suite du TP, on notera $F_{i/j}$ le mot qui est la concaténation de F_i et F_j (dans cet ordre) ayant en commun $\text{overlap}(F_i, F_j)$ lettres. On définit $\text{overlap}(F_i, F_j)$ la longueur du plus long suffixe de F_i qui correspond à un préfixe de F_j . On cherche donc la valeur maximale dans ce tableau :

	F_1	F_2	F_3	F_4	F_5
F_1		$\text{overlap}(F_1, F_2)$	$\text{overlap}(F_1, F_3)$	$\text{overlap}(F_1, F_4)$	$\text{overlap}(F_1, F_5)$
F_2	$\text{overlap}(F_2, F_1)$		$\text{overlap}(F_2, F_3)$	$\text{overlap}(F_2, F_4)$	$\text{overlap}(F_2, F_5)$
F_3	$\text{overlap}(F_3, F_1)$	$\text{overlap}(F_3, F_2)$		$\text{overlap}(F_3, F_4)$	$\text{overlap}(F_3, F_5)$
F_4	$\text{overlap}(F_4, F_1)$	$\text{overlap}(F_4, F_2)$	$\text{overlap}(F_4, F_3)$		$\text{overlap}(F_4, F_5)$
F_5	$\text{overlap}(F_5, F_1)$	$\text{overlap}(F_5, F_2)$	$\text{overlap}(F_5, F_3)$	$\text{overlap}(F_5, F_4)$	

Étape 1 :

	$F_1 : ACCTGAG$	$F_2 : TGCATTGC$	$F_3 : GCAGACC$	$F_4 : AGCAAT$	$F_5 : CAATG$
$F_1 : ACCTGAG$		0	1	2	0
$F_2 : TGCATTGC$	0		2	0	1
$F_3 : GCAGACC$	3	0		0	1
$F_4 : AGCAAT$	0	1	0		4
$F_5 : CAATG$	0	2	1	0	

On remarque que le plus gros overlap est $\text{overlap}(F_4, F_5) = 4$ (les 4 dernières lettres de F_4 sont égales aux 4 premières de F_5). On obtient donc le nouveau mot $F_{4/5} = AG\underline{CAAT}G$ (la séquence d'overlap est soulignée) et on supprime F_4 et F_5 .

Étape 2 :

	$F_1 : ACCTGAG$	$F_2 : TGCATTGC$	$F_3 : GCAGACC$	$F_{4/5} : AGCAATG$
$F_1 : ACCTGAG$		0	1	2
$F_2 : TGCATTGC$	0		2	0
$F_3 : GCAGACC$	3	0		0
$F_{4/5} : AGCAATG$	0	2	1	

On remarque que le plus gros overlap est $overlap(F_3, F_1) = 3$ (les 3 dernières lettres de F_3 sont égales aux 3 premières de F_1). On obtient donc le nouveau mot $F_{3/1} = GCAGACCTGAG$ (la séquence d'overlap est soulignée) et on supprime F_3 et F_1 .

Étape 3 :

	$F_{3/1} : GCAGACCTGAG$	$F_2 : TGCATTGC$	$F_{4/5} : AGCAATG$
$F_{3/1} : GCAGACCTGAG$		0	2
$F_2 : TGCATTGC$	2		0
$F_{4/5} : AGCAATG$	1	2	

Dans notre procédure gloutonne, on suppose que l'overlap qui nous intéresse est celui que l'on rencontre en premier (en parcourant le tableau dans l'ordre ligne puis colonne).

Cet overlap est $overlap(F_{3/1}, F_{4/5}) = 2$ (les 2 dernières lettres de $F_{3/1}$ sont égales aux 2 premières de $F_{4/5}$). On obtient donc le nouveau mot $F_{(3/1)/(4/5)} = GCAGACCTGAGCAATG$ (la séquence d'overlap est soulignée) et on supprime $F_{3/1}$ et $F_{4/5}$.

Étape 4 :

	$F_{(3/1)/(4/5)} : GCAGACCTGAGCAATG$	$F_2 : TGCATTGC$
$F_{(3/1)/(4/5)} : GCAGACCTGAGCAATG$		2
$F_2 : TGCATTGC$	2	

D'après notre procédure gloutonne (vue précédemment), l'overlap qui nous intéresse est $overlap(F_{(3/1)/(4/5)}, F_2) = 2$ (les 2 dernières lettres de $F_{(3/1)/(4/5)}$ sont égales aux 2 premières de F_2). On obtient donc le nouveau mot $F_{((3/1)/(4/5))/2} = GCAGACCTGAGCAATGCAATTGC$ (la séquence d'overlap est soulignée) et on supprime $F_{(3/1)/(4/5)}$ et F_2 .

Finalement, l'algorithme renvoi la séquence (de longueur 22) :

$$F_{((3/1)/(4/5))/2} = F = \mathbf{GCAGACCTGAGCAATGCATTGC}$$

Exercice 2

Algorithme 1 : Assemblage($F = \{F_1, F_2, \dots, F_n\}$: ensemble de mots) : superchaîne

Variables : SC: ensemble de mots; max : entier; a, b : mot;

$SC \leftarrow F$;

$max \leftarrow 0$;

$a \leftarrow \text{""}$;

$b \leftarrow \text{""}$;

tant que $|SC| > 1$ **faire**

pour chaque mot m_1 de SC **faire**

pour chaque mot m_2 de $SC \setminus \{m_1\}$ **faire**

si $max == 0$ **alors**

$a \leftarrow m_1$;

$b \leftarrow m_2$;

fin

$buf \leftarrow overlap(m_1, m_2)$;

si $max < buf$ **alors**

$max \leftarrow buf$;

$a \leftarrow m_1$;

$b \leftarrow m_2$;

fin

$SC \setminus \{m_1\} \cup \{m_2\}$;

$SC \leftarrow sousMot(m_1, max) + m_2$; /* sousMot est une fonction qui prend en argument un mot et un entier n et renvoie le mot privé de ses n derniers caractères.

 */

fin

fin

fin

renvoyer SC;

Complexité: $o((n^3) * o(overlap(m, m)))$ avec n le nombre de mots et m la taille du mot le plus grand. C'est algorithme pourrait être amélioré en gardant en mémoire les overlaps par pair de sommet afin d'éviter de les recalculer; l'algorithme serait alors en $o((n^2) * o(overlap(m, m)))$

Exercice 3

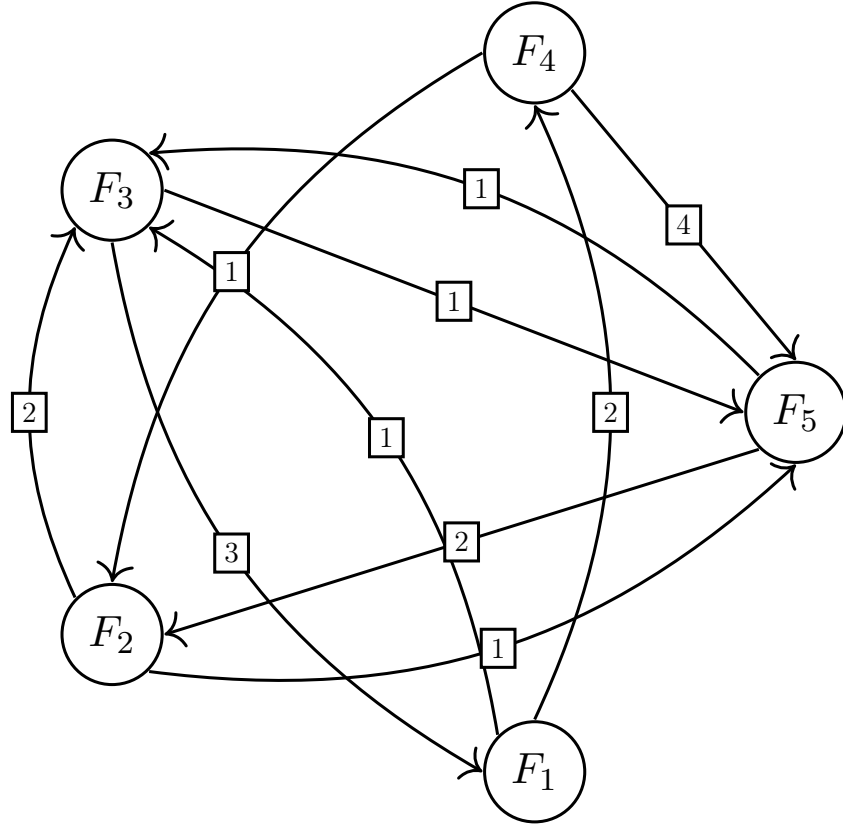


FIG. 1 – Graphe de chevauchement de l'Exercice 1

Algorithme 2 : $\text{creationGraphe}(F = \{F_1, F_2, \dots, F_n\} : \text{ensemble de mots}) : \text{graphe } G(V, A)$

Variables : $G(V, A)$: un graphe;

$V \leftarrow F$;

$A \leftarrow \{\}$;

pour chaque mot m_1 de V **faire**

pour chaque mot m_2 de $V \setminus \{m_1\}$ **faire**

$\text{buf} \leftarrow \text{overlap}(m_1, m_2)$;

si $\text{buf} > 0$ **alors**

$A \leftarrow A \cup \{m_1, m_2, \text{buf}\}$;

fin

fin

fin

renvoyer G ;

Complexité: $o((n^2) * o(\text{overlap}(m, m)))$ avec n le nombre de mots et m la taille du mot le plus grand.

Exercice 4

Algorithme 3 : cheminHamiltonien($G(V,A)$: graphe de chevauchement) : chemin hamiltonien

Variables : L: liste de sommets; max: entier; savV: ensemble de sommets; savS: sommet; savA: arête

```

L ← {};
savV ← V;
savS ← null;
tant que |savV| > 0 faire
    max ← 0;
    savA ← null;
    pour chaque arête a de A faire
        si a[2] > max et a[1] ∈ savV et a[0] = savS alors
            max ← a[2];
            savA ← a;
        fin
    fin
    si savS = null alors
        savV ← savV \ {savA[0]};
    fin
    savV ← savV \ {savA[1]};
    L ← L ∪ {savA[0]} ∪ {savA[1]};
    savS ← savA[1];
fin
renvoyer L ∪ L[1];

```

Complexité: $O(n * m)$ avec n le nombre de sommets et m le nombre d'arêtes.

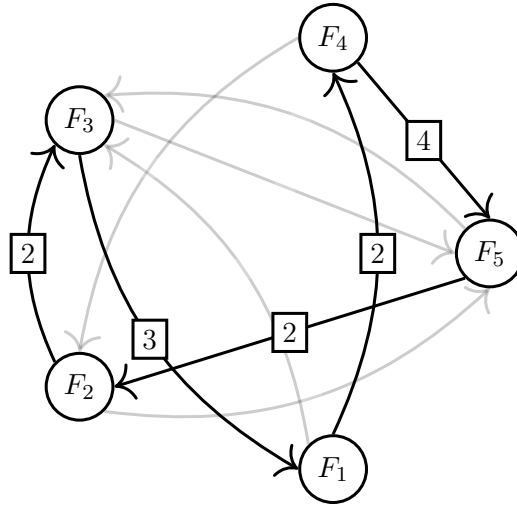


FIG. 2 – Chemin hamiltonien du graphe de chevauchement de l'**Exercice 1**

Le chemin hamiltonien vaut donc 13 et on peut reconnaître le motif trouvé à l'exercice 1.