M1 - BIF
TP3 - Transformée de Burrows Wheeler

Victor Levallois, Claire Lemaitre, Pierre Peterlongo

2023-2024

1 Build BWT

Q 1. Codez (et testez) la fonction $get\_BWT(S, SA)$ renvoyant la transformée de Burrows Wheeler BWT du texte S. Pour le calcul du tableau des suffixes, vous pouvez utiliser votre implémentation du TP2 ou utiliser la fonction SA = simple_kark_sort(S). (from karkkainan import simple_kark_sort)

2 Reverse BWT

Q 2. Codez la fonction $get\_R(BWT)$ renvoyant un tableau tel que $R[i]$= rang dans BWT du caractère $BWT[i]$.

Q 3. Codez la fonction $get\_N(BWT)$ renvoyant un tableau associatif tel que $N[c]$ = position du premier c dans F.
Rappel : si BWT(S) est la dernière colonne de la matrice des rotations cycliques de S dans l'ordre lexicographique, F en est la première colonne.

Q 4. Codez la fonction $LF(c, r, N)$ qui renvoie la position dans F du caractère c de rang r.

Q 5. Codez la fonction $BWT2seq(BWT, N, R)$ qui reconstruit, en temps linéaire, la séquence S à partir de sa BWT.

3 Pattern matching - BWT

Q 6. Codez la fonction $find\_first(c, i, BWT)$ qui renvoie la position du premier c entre i et $|BWT| - 1$.

Q 7. Codez la fonction $find\_last(c, j, BWT)$ qui renvoie la position du dernier c entre 0 et j.

Q 8. A partir des fonctions LF, $find\_first$ et $find\_last$, codez la fonction $P\_in\_S(P, BWT, N, R, SA)$ qui renvoie les occurences de P dans S si elles existent, sinon -1.

4 Bonus

Q 9. Garder les rangs est coûteux en mémoire, implémentez un sous-échantillonage des rangs pour sacrifier un peu de temps de calcul au profit d'espace mémoire. Pour ce faire , codez $get\_R\_bis(BWT, p)$ qui notera une valeur des rangs de chaque caractère de la BWT tous les p caractères.

Q 10. Idem, pour SA.