Bonjour et bienvenue dans cette présentation pour le cours de MOOC, alors moi j’ai choisi de faire la bio-informatique et j’ai choisi de faire la présentation en anglais parce que les cours était dispenser en Anglais, mais l’exposer sera en français.

Pour notre plan, nous avons Overvieuw où nous allons voir les partis de la bio-informatique pour la session dont j’ai été inscrit. La présentation en bref des deux algorithmes qui nous a été demandé de choisir pour notre présentation. Et évidemment le choix que nous avons fait.

Dans l’introduction, nous verrons une définition de la bio-informatique et où l’informaticien est impliqué.

Dans la deuxième partie, c’est plutôt tout ce qui concerne l’algorithme que nous avons choisi. Et aussi un petit peu sur l’autre l’algorithme.

Dans la troisième partie ça sera le problème et la résolution de notre algorithme, ainsi qu’une explication de l’algorithme.

Nous avons eu trois parties et le langage que nous avons utilisé pour coder c’est le python la version 2.7.13.

La bio-informatique est un domaine de recherche qui ressemble plusieurs disciplines à savoir : la biologie, la médecine, l’informatique, la mathématique, la physique. Les personnes travaillent ensemble dans un problème scientifique posé par la biologie

Dans tout ça c’est quoi le rôle d’un informaticien : vu que la bio-informatique veut accroître la compréhension des processus biologiques, alors pour ce faire ils ont besoin de programme pour leur aider à bien résoudre et rapidement leur problème. C’est qui distingue la bio-informatique des autres approches, c’est qu’elle met l’accent sur le développement et l’application de techniques intensives de calcul pour attendre l’objectif.

Parlons maintenant de l’algorithme que nous avons choisi. Il se trouve dans quel parti et dispenser par qui ?

Le deuxième algorithme que nous allons par présenter est ‘le trait changeant de l’alignement de séquences’

Commençons par quelques définitions pour bien comprendre le théorème d’Euler. Définissons chemin eulérien qui est différent d’un cycle eulérien. Comme expliquer dans le graphe qui va suivre.

Le théorème d’Euler : un graphe connexe est Eulerien si et seulement si tous ses nœuds sont de degré pair. Si G est un chemin d’Euler, il doit avoir exactement deux sommets impairs.

Voyons maintenant notre problème :

Explication du code :

* Le dictionnaire à l’arête contient les lignes déjà lues. Le début comme une touche et la destination comme une liste. On crée une boucle pour cela
* Après terminé l’interprétation des données, un appel est fait à la fonction eulerian\_cycle. La liste Path est générée et contiendra eurlerian\_cycle(arêtes). Il y a l’intervention du dictionnaire pour obtenir le cycle initial. Nous mettons current\_node à la première clé dans le dictionnaire edge\_dict.
* La première instruction dans la boucle while ajoute la première valeur pour laquelle la clé est le current\_node à la liste de chemins. La première condition if dans la boucle supprime la clé et la valeur du dictionnaie edge\_dict s’il y a qu’une seule valeur, ou elle remplace la première valeur de cette clé par la valeur suivante
* Met à jour curren\_node avec la valeur ajoutée la plus récente dans Path si cette valeur existe comme une clé dans edge\_dict, sinon elle casse la boucle. Si la mise à jour est réussie, cela signifie qu’un cycle initial a été créée mais elle est incomplet est trouvé et on suppose c’est le cas.
* Pour la deuxième partie de la fonction, nous définissons la condition suivante ; si la longueur du dictionnaire edge\_dict est supérieur à 0, nous étendons continuellement le cycle initial jusqu’à ce que nous soyons hord de edge\_dict.
* Nous créons la liste appelée cycle. En utilisant les mêmes principes que dans la première partie de la fonction, nous définissons la première valeur de cette liste sur le chemin de valeur[i]. Ensuite, nous utilisons une version légèrement modifiée de la boucle While True. La liste des cycle est alors attachée au chemin , en utilisant les instructions
* Quand et si nous atteignons une autre impasse, l’algorithme est exécuté à lorsque jusqu’à ce que edge\_dict soit vide, Path contient le cycle que nous recherchons.