JNI et affichage d'alignements sous forme de dotplot ou de texte

Nous disposons maintenant d'une bibliothèque (même si elle ne dispose que d'une seule fonction) avec une fonction de calcul d'alignement global entre deux séquences. Nous voulons créer une interface graphique en Java dont la fonctionalité est le calcul et l'affichage d'un alignement global entre deux séquences. Afin de disposer d'un outil de comparaison d'alignements, nous allons créer une application fenêtrée qui permettra d'afficher simultanément les réultats de plusieurs alignements. Les alignements pourront être vus selon deux modes : soit dans un mode texte, soit dans un mode graphique. Pour le mode texte, on écrira simplement l'alignement dans une zone de texte. Pour le mode graphique, on fera un affichage à la dotplot dans un panel.

1 Les techniques employées

1.1 JNI

Pour utiliser des fonctions d'une bibliothèque C en Java nous allons utiliser les JNI (Java Native Interface). A partir d'une classe Java dans laquelle sont déclarées des méthodes dites natives, l'utilitaire `javah -jni` permet de produire le fichier d'entête C correspondant aux méthodes natives. Ensuite, il suffit de créer le fichier de corps C qui peut appeler n'importe qu'elle fonction écrite en C. Bien sûr, une des principales difficultés est l'échange des paramètres entre Java et C. Le tutoriel à l'adresse `http://java.sun.com/docs/books/tutorial/native1.1/stepbystep/index.html` permet de bien comprendre le fonctionnement.

Exercices

- Créez une bibliothèque C contenant une méthode `void toto () ;` qui imprime "Ca marche!" à l'écran.
- Créez un programme test java qui possède une méthode native correspondant à `print`
- Compilez pour créer votre premier programme Java utilisant une fonction C.
- Modifiez votre code pour que `void print (toto *str)` imprime le contenu de `str`
- Modifiez votre code pour que `char *toto (toto *str)` imprime le contenu de `str` puis retourne ce que l'utilisateur a entré.
- Comment feriez-vous pour récupérer la structure ou les éléments de la structure `alignment` retournée par la fonction `nw` ?

1.2 Swing

Pour la création de l'application graphique, nous allons utiliser l'API Swing fournie par Java. Cetta API permet le développement rapide d'applications graphiques. En particulier, il est facile de créer une application multifenêtrée dont les fenêtres sont toutes sur un "bureau". L'application principale doit alors créer un `JDesktopPane` qui peut alors lui-même contenir des `JInternalFrame` qui seront les fenêtres.

Pour découvrir et comprendre comment cela fonctionne regardez le code de l'exemple fourni sur la page web habituelle. Regardez aussi le tutorial (dont l'exemple est une émanation) :
`http://java.sun.com/docs/books/tutorial/uiswing/index.html` et le tutoriel Swing :
`http://java.sun.com/docs/books/tutorial/uiswing/start/index.html`.

1.3 Observateur / Observable

Maintenant, il faut savoir comment faire communiquer une interface graphique représentant un objet et l'objet lui-même. Pour cela, nous allons utiliser le paradigme observateur/observable. L'idée est simple : nous déclarons un objet comme étant observable. Cela veut dire qu'il est possible de dire à cet objet qu'il possède des observateurs. Lorsque l'objet est modifié, celui-ci envoie un message à ses observateurs qui peuvent alors effectuer le rendu graphique voulu. L'observateur connaît bien sûr l'objet qu'il observe, il suffit donc de mettre en place des méthodes sur l'objet qui permette d'obtenir ses composants. Ainsi, il y a séparation complète entre l'écriture du code de l'objet et son rendu.

Rendre une classe observable se fait en implémentant l'interface `Observable`. Créer un observateur se fait en implémentant l'interface `Observer`.

Exercices

- Proposez une classe `Alignment` qui sauve un alignement et qui est observable.
- Créer deux observateurs, l'un graphique à la dotplot, l'autre textuel, permettant de visualiser des alignements.

2 A votre disposition

Pour comprendre le fonctionnement des interfaces graphiques, un exemple est fourni, suffisamment proche de ce qui vous est demandé pour que le travail à fournir soit simple et rapide :

`Xdesktop.java`	l'application principale
`Xalignment.java`	sous fenêtre (internal frame) affichant soit un dotplot soit un alignement texte
`DotplotAlignmentViewer.java`	un exemple d'affichage graphique
`TextAlignmentViewer.java`	un exemple d'affichage en mode texte

Pour lire des séquences et des matrices, des classes sont fournies (que vous devrez utilisez) :

`Alphabet.java`	représente un alphabet
`Sequence.java`	représente une séquence
`SequenceReader.java`	interface fournissant un patron pour l'implémentation de classes de lecture de fichiers de séquences
`FastaSequenceReader.java`	implantation de SequenceReader pour des fichiers Fasta contenant plusieurs séquences (attention case sensitive)
`Matrix.java`	représente une matrice
`Matrixreader.java`	interface fournissant un patron pour l'implémentation de classes de lecture de fichiers de matrice
`RectangularMatrixReader.java`	implantation de MatrixReader pour des fichiers de matrices rectangulaires au format du premier TP

En plus de cela deux programmes de test : `TestSequence.java` et `TestMatrix.java` à utiliser avec les fichiers `ltr.fasta` et `mat`.

3 Ce qui est à rendre (pour le 10 décembre)

Une application multifenêtrée permettant :
- de saisir des séquences à partir d'un fichier fasta contenant plusieurs séquences à la suite,
- d'ajouter des matrices à partir de fichiers,
- de régler des gaps et des gops
- de lancer le calcul pour un jeu de score et d'afficher le résultat selon un alignement

Rendre un jar contenant un README, un Makefile, les sources C et Java. Vous utiliserez la javadoc pour commenter votre programme.