Docking Moléculaire

Master 1 I2L - 2014 / 2015
A rendre avant le jeudi 4 décembre minuit.

1 Description

Le docking moléculaire consiste à déterminer la position relative de deux molécules (un ligand et un récepteur). La structure obtenue confère les propriétés à l'ensemble (le complexe) ainsi formé. Par exemple, la recherche de 'bonne' structure est cruciale dans la conception de nouveau médicament. Ou encore, selon l'association de 2 protéines, le signal déclenché lors de leur association peut être différent.

D'un point de vue informatique, la prédiction de structure se traduit par un problème d'optimisation. A chaque position relative possible est associé une énergie (également appelé score, fitness, etc.). La structure la plus probable est alors celle qui minimise cette énergie.

2 But

Proposer et étudier des algorithmes d'optimisation pour le problème de docking.

3 Représentation du problème

Pour réaliser ce projet, un code c++ vous ait fournit. Le code donne le calcul de l'énergie d'une position relative entre le récepteur et le ligand. Ce code fait partie de la bibliothèque PTools en partie développée par Sébastien Fiorucci de l'Université Nice Sophia Antipolis. Vous trouverez une archive *docking.zip* contenant ce code avec plusieurs exemples. Pour compiler et exécuter les exemples, veuillez suivre les instructions décrites dans le fichier `README.txt`.

4 Travail Demandé

Il vous est demandé de réaliser plusieurs méthodes d'optimisation (cf. questions). Vous testerez chaque méthode sur le benchmark de complexes fourni dans le dossier `data`. Pour vous aidez, un script `run.sh` permet de lancer une campagne d'expérimentation et d'obtenir la performance d'une méthode sur l'ensemble du benchmark.

Dans votre rapport, on attend la description précise des algorithmes, du protocole expérimental de test, un rapport statistique qui permet de comparer les algorithmes, ainsi qu'une analyse des résultats obtenus.

Questions :

a - Développer et tester un algorithme $(1 + 1)$-ES (fichier `src/minimizers/onePlusOneES.h`).

b - Développer et tester un algorithme $(1 + 1)$-ES avec la règle 1/5 success rule (fichier `src/minimizers/onePlusOneFifthRuleES.h`).

c - Développer et tester un algorithme $(\mu/\mu, \lambda)$-ES (fichier `src/minimizers/muSlashMuCommaLambdaES.h`).

d - Proposer et étudier une extension possible des algorithmes précédents, par exemple en proposant une méthode d'initialisation particulière, une méthode de 'restart' particulière, ou encore en utilisant une bibliothèque qui permet de tester l'algorithme de CMA-ES (voir liens utiles).

5 Lien utiles

— Docking moléculaire : `http://en.wikipedia.org/wiki/Docking_(molecular)`
— Concours CAPRI : `http://www.ebi.ac.uk/msd-srv/capri/`
— métaheuristiques : `http://fr.wikipedia.org/wiki/Mtaheuristique`
— stratégies d'évolution : `http://fr.wikipedia.org/wiki/Stratgies_d%27volution`
— PTools : `http://ptoolsdocking.sourceforge.net/`
— Code du CMA-ES : `https://www.lri.fr/~hansen/cmaes_inmatlab.html`
— Librairie c++ boost : `http://www.boost.org/`