2009 年3月 17 日

非数値計算部門課題

非数値計算部門の課題は、「相同性検索プログラムの並列化・高速化」です。

相同性検索プログラムとは

相同性検索プログラムは、バイオインフォマティックス分野で生物配列（塩基配列やアミノ酸配列）間の類似性を調べるためのプログラムです。一般的には、既知の配列群をデータベース、新しく見つかった配列群をクエリーとし、クエリー配列と類似の配列をデータベースから探すときに使用されます。

競技方法

相同性検索プログラムのサンプルプログラムを提供しますので、このサンプルプログラムを並列化・高速化して下さい。並列化・高速化したプログラムの起動から終了までの経過時間を競って頂きます（経過時間は time コマンドで計測）。
並列化・高速化手法に関しては特に制限をつけません。何をして頂いても OK ですが、サンプルプログラムと同じ出力が得られることを正解の条件とします。ただし、本コンテストで使用する相同性検索アルゴリズム（Smith-Waterman アルゴリズム）では、最適な検索結果が複数存在する可能性がありますので、サンプルプログラムと出力が異なる場合でも、それがSmith-Waterman アルゴリズムとして最適な検索結果であれば正解とします。なお、不正解の場合には、失格・減点となります。
予選・本選の具体的な実施方法は（プログラム提出方法・実行方法・配点等）、決まり次第参加者に連絡します。

サンプルプログラムの配布場所

サンプルプログラムは下記 URL にて配布します。ファイル名は sw-sample.(日付).tgz です。

https://www2.cc.u-tokyo.ac.jp/procon/data/homology/

サンプルプログラムの実行方法

サンプルプログラムの実行方法を説明します。
まず、ダウンロードした tgz ファイルを展開して make します。

```
$ tar　xfz　sw-sample.(日付).tgz
$ cd　sw-sample.(日付)
$ make
```

データセットは練習用・予選用に１１セット用意してあります（p0～p5、q1～q5）。以下のようにデータセット名を指定してスクリプト run.sh を起動すると、指定データセットに対する検索が実行されます。

```
$ ./run.sh　p0
```

実行結果は指定データセット名のディレクトリ内に出力されます（上記例だとディレクトリ p0 内）。検索結果はファイル output.(日付)に、時間計測結果はファイル log.(日付)に出力されます。

サンプルプログラムの挙動

サンプルプログラムのおおまかな挙動は以下の通りです。

1. スコア行列ファイルの読み込み（load_score_matrix()）
2. クエリーファイルの読み込み（load_sequence_set()）
3. データベースファイルの読み込み（load_sequence_set()）
4. クエリー配列を 1 つ選択
5. 選択したクエリー配列と、各データベース配列との類似度を計算（get_smith_waterman_score()）
6. 最も類似度の高いデータベース配列とのアライメントを計算・出力（show_alignment()）
7. 未処理クエリー配列がある場合は 4.に戻る
8. おわり

配列間の類似度計算には Smith-Waterman アルゴリズムを使用しています。Smith-Waterman アルゴリズムの詳細は Web 等で検索して下さい。

参考 URL

http://www.bi.a.u-tokyo.ac.jp/~shimizu/bioinfo/homology.html
http://trans.nsc.nagoya-cu.ac.jp/~mano/bio/part2/part2.html
http://www.cis.nagasaki-u.ac.jp/~masada/DA2008/7.html

データセット

データセット（スコア行列ファイル、クエリーファイル、データベースファイル）は、練習用が 6 セット（p0～p5）、予選用が 5 セット（q1～q5）の計１１セットが用意されています。

- p0： とても小規模なデータセットです。サンプルプログラムでも数十秒で検索が完了します。プログラムの動作確認にはこのテストセットを使うことをお勧めします。
- p1～p5： 小規模なデータセットです。サンプルプログラムで 15～30 分程度かかります。
- q1～q5： 中規模なデータセットです。サンプルプログラムで 100～150 分程度かかります。予選にはこのテストセットを使用します。このテストセットは、プログラムの並列化・高速化がある程度進んでから使うことをお勧めします。

各データセットにはサンプルプログラムの出力例が付いています（ファイル名:output.sample）。プログラムを並列化・高速化したら、出力がこれと一致するかどうかを確認して下さい。
なお、本選ではデータセットが変わります。本選用のデータセットは事前公開しません。

プログラムの出力形式

検索結果は以下形式にて出力されます。

```
Query sequence: 1_DS4kVs8niKrPkEKLgbe4
Database sequence: 10_vublYBsRWOlvlg1CNQ5E
Best score: 54
Q:     4 VQYDCDVAREGAGSVAEIFYVLNLTTVSKIEW 35
D:   186 VRLHVDITRDGKTVVSEIYVDDNLLTNKKDQW 217
```

1 行目: クエリー配列名

２行目: データベース配列名
３行目: クエリー配列とデータベース配列の類似度
４行目: アライメント結果（クエリー配列）
５行目: アライメント結果（データベース配列）

あるクエリー配列に対して、最も類似度の高いデータベース配列が複数あるときは、それらデータベース配列全ての検索結果が出力されます。そのときの出力順は、データベース内の配置順となります（データベースファイル内にて先に記載されている配列の検索結果が先に出力される）。プログラムを並列化・高速化した後も、この出力順を守る必要があります。出力順が異なる場合は失格・減点の対象となります。

配列ファイルの形式と配列番号

データセット内の配列ファイル（クエリーファイル、データベースファイル）は FASTA 形式と呼ばれる形式を使用しています。FASTA 形式では複数の配列を記述することができ、「>」で始まる行が各配列の名前を表し、それに続く行がその配列の成分を表します。以下、FASTA 形式により２本の配列を記述した例です。

```
>1_abc                  # １本目の配列名
MPAAMLNSGEALACQ         # １本目の配列成分
FTGIIKTFFGEVAMNCA       # １本目の配列成分
>2_def                  # ２本目の配列名
NGLSQTWQAGEVNLL         # ２本目の配列成分
```

本コンテストで使用する配列ファイルにおいては、配列名は必ず以下形式に従うものとします。

配列名の形式：(配列番号)_(任意の文字列)

配列番号は、ファイル内における配列の順番（昇順、１始まり）を示す番号です。検索結果の出力順をソートする必要がある場合、この配列番号に依存してソート実装しても、本選テストセットでも問題なく動作することを保証します（もちろん、そのソート実装にバグの無いことが前提）。

並列化・高速化に関して

相同性検索プログラムの並列化・高速化には色々なアプローチが考えられます。以下、代表的なアプローチを簡単に説明しますので参考にして下さい。

- クエリー分配
  クエリーを配列単位で各 CPU コアに分配し、各 CPU コアは別個のクエリー配列に対して相同性検索する方法です。並列化というより分散化と呼ぶのが適切な方法で、並列化オーバーヘッドは非常に小さいです。クエリー配列数が多い場合には有効ですが、クエリー配列数が少ない場合は効果が限定されます。

- データベース分割
  データベースを配列単位で分割し、各 CPU コアは別個のデータベースから類似配列を検索する方法です。各 CPU コアの検索結果をまとめる必要があるので、若干の並列化オーバーヘッドがあります。データベース配列数が多い場合には有効ですが、データベース配列数が少ない場合は効果が限定されます。

- 配列のブロック化
  配列をあるブロック単位で分割して検索する方法です。配列長が長い場合には有効な方法ですが、上記 2 手法と比べると並列化オーバーヘッドが大きく、また配列長が短いときには並列化効果を得るのが困難です。

正解の条件

基本的にはサンプルプログラムと同じ出力が得られることが正解の条件ですが、Smith-Waterman アルゴリズムでは最適な検索結果が複数存在する可能性がありますので、サンプルプログラムと出力が異なる場合でも、それがSmith-Waterman アルゴリズムとして最適な検索結果であれば正解とします。
以下に最適な検索結果が複数存在する具体例を示しますので、参考にして下さい。

- ベストスコアが複数存在する例

```
       1                   20
配列 1: ABCDEVVVVVVVVVVEDCBA
配列 2: ABCDEWWWWWWWWWWEDCBA

[正解 1]
   1 ABCDE 5
   1 ABCDE 5

[正解 2]
   16 EDCBA 20
   16 EDCBA 20
```

- 最適なアライメントが複数存在する例

```
       1     8
配列 1: ABCDDEFG
配列 2: ABCDEFG

[正解 1]
   1 ABCDDEFG 8
   1 ABCD-EFG 7

[正解 2]
   1 ABCDDEFG 8
   1 ABC-DEFG 7
```

以上