COMP90025 Parallel and Multicore Computing

Project 2A - Sequence Alignment

Aaron Harwood and Lachlan Andrew

School of Computing and Information Systems
The University of Melbourne

2020 Semester II

Summary

- This project is individual and is divided into two parts. This is Part A.
- A sequential algorithm, in C++ but practically C, for computing the alignment of two sequences has been provided on Canvas. The algorithm is adapted from this website: `https://www.geeksforgeeks.org/sequence-alignment-problem/` which also has some supporting discussion.
- The sequence alignment problem is roughly, given two sequences of symbols, insert gaps into the sequences such that the penalty, that is a function of how many gaps are inserted and mis-matches in symbol alignment, is minimized, when the two sequences are compared symbol for symbol at each index location. This problem is commonly attributed to Bioinformatics where it is concerned with aligning sequences of amino acid base pairs, or gene sequences. However sequence alignment has many varied applications.

Input format

- The algorithm given reads the problem from standard input, and an example problem is also supplied. It can be run like:

  ```
  g++ seqalign.cpp
  cat seq.dat | ./a.out
  ```

- The first line of input is the mis-match penalty as an integer.
- The second line of input is the gap penalty as an integer.
- The third line of input is the first sequence as a string of character symbols.
- The fourth line of input is the second sequence as a string of character symbols.

```
3
2
AGGGCT
AGGCA
```

Tasks

- Write an OpenMP program that computes and outputs, in the same way, the sequence alignment as done by the sequential program.
- Aim to have your program run as fast as possible. In doing so, you may alter the calculations of the program, so long as the final output is correct.
- Use the sequential code as a base. Do not change the file before the end of `main()`. If you need to include additional headers, they can be included after `main()`. In particular, do not alter the timing statements or add any of your own timing statements.
- Write up to 200 words that briefly discusses the parallel techniques used and cite any sources that you referred to when implementing your approach, e.g. research papers that talk about parallel solutions to the sequence alignment problem.

Assessment

- Project 2 A is worth **10%** of your total assessment. The written submission is 2/10 and the program assessment is 8/10.
- Assessment of the program is based on correctness and performance relative to the performance of all submissions from the class. Incorrect programs (i.e. that give incorrect outputs or that fail to compile/run) will attract few if any marks. The top 5 fastest running programs, when given one or more mystery problem instances (you will not be told the actual test problem instances in advance), will be given a bonus mark; i.e. the maximum mark for this project part is 11/10.
- At most five people will get the bonus mark. If a sixth person is tied with anyone in the top five, then anyone tied with that sixth person will not receive a bonus mark.

Submission

- Project 2 A is due **Friday September 11th, 23:59**.
- The written submission will be an assignment submission via LMS, similar to other written submission for this subject. As well you will need to submit your program (either via LMS or directly on Spartan). Instructions for doing this will be given closer to the deadline.
- Optimal code often requires specific compiler options. As a comment in the last line of your C code, put the exact command line used to compile your code. Do not put anything else on that line.