Calculating genomic coverage from next-generation sequencing data

Overview

Our team at the MGH Center for Integrated Diagnostics performs tumor genotyping for cancer patients at MGH, with the aim of detecting abnormalities which can then guide personalized therapy. Our results are routinely used in cancer care at MGH, often being the last hope for people with Stage III and Stage IV tumors for whom traditional therapies have failed.

In this puzzle, we ask you to solve a real-world DNA sequencing problem. Don't worry, we assume no biology/bioinformatics experience. We give you a couple of days for flexibility, but we definitely don't expect you to spend more than a couple of hours total working on it. Most of all, we hope you have fun!

Deliverables

We are looking for solutions that are correct, creative and cleanly implemented. Bare-metal performance is important but isn't our primary concern -- we will favor slower solutions that are readable and well-documented, over ones that are fast but incomprehensible. Likewise, we will favor solutions with a better algorithmic complexity even if your specific implementation isn't particularly fast (e.g. custom search algorithm in Python is better than a naive filter in pandas, even if the latter is faster for the data sizes in this puzzle).

At the end of the exercise, we ask you to send us a single archive containing:

1. Your code
2. The loci.csv file with the coverage column filled in
3. A short writeup (1 page is sufficient) describing your overall approach, and any optimizations you implemented or considered.

Please try to be creative with your solution. Many people will implement their answer as a simple `for` loop which, while correct, will not make your answer stand out from the crowd. Try to think of algorithmic or insightful ways to optimize the problem:

- Is there a way to search through the reads faster than O(n)?
- The reads aren't random. You might be able to preprocess them in a way that considerably reduces your search space.
- etc.

Please send your solutions to mpacula@partners.org.

Puzzle

The key to what we do at CID is called Next Generation Sequencing (NGS), a relatively new sequencing approach which works by breaking up the DNA into short fragments, replicating them in large quantities, then sequencing those short fragments in a high-throughput fashion. The output of an NGS sequencer is a large collection of such disjointed sequences, called "reads". For the purpose of this puzzle, each read has two properties: (1) a start position within the genome and (2) its length. Both properties are expressed in base pair units, i.e. letters of DNA like A,T,C,G. For example, the read "AAATCGA" has length 7.

A key component of making sense of NGS data is calculating "coverage" (also known as "read depth") at a given genomic position. At its most basic, coverage is simply the number of reads overlapping a position in the genome. High coverage gives a measure of confidence in the sequencing results, and the calculation of coverage is a critical component of our software systems at MGH CID. In this exercise, you will work with a real sequencing output to calculate read coverage at a number of positions of interest ("loci").

Data

You have two files:

- `reads.csv`: this file contains approximately 2mln reads, one read per row, with two columns corresponding to the start position and length of the read. For example, in the excerpt below, the first read starts at position 101843359 and has a length of 151 base pairs.

```
start,length
101843359,151
101891952,151
148529640,139
101921163,42
139044920,150
151860131,150
140476640,147
50358597,150
101813657,150
```

- `loci.csv`: this file contains 1000 positions of interest and a blank "coverage" column. In the course of this exercise, you will populate the coverage column with the number of reads overlapping that position.

```
position,coverage
101844980,
104748289,
101892012,
101870868,
148512562,
101921050,
101891648,
101891584,
101892180,
```

The data came from a real heme myeloid case sequenced at the CID.

The exercise is to write a computer program in a language of your choice (Python, Java or C/C++/C# preferred) to calculate the coverage for each of the positions in loci.csv, based on the reads in reads.csv. Coverage is defined as the number of reads which overlap that position.

Examples

Suppose you have the following 2 reads:

```
start,length
10,30
20,40
```

Now suppose we're interested in the coverage at 3 loci: 5, 15, and 30. None of our reads overlap 5, so the coverage there is 0. Only the first read overlaps 15, so the coverage at 15 is 1. Both reads overlap 30, giving a coverage of 2:

```
position,coverage
5,0
15,1
30,2
```

Expected output

As a sanity check for your implementation, here's the expected output for the first 20 loci in loci.csv:

```
position,coverage
101844980,1190
104748289,12206
101892012,141295
101870868,6809
148512562,6993
101921050,1096
101891648,62732
101891584,43740
101892180,78510
101891581,42854
148529493,62
139044861,221
101921385,24560
151901886,1
151878406,2581
139102584,84
101840276,6209
101877167,36
101920996,431
139044978,213040
```

Bonus question

Please measure run-time performance of developed code in terms of numbers of lines in reads.csv and loci.csv. As additional optional challenge describe ways of improving performance if you see any.