**BIOL4292: Programming and Databases CW1**

**Script Feedback**

**Does the script run without errors?**

Yes

**Does the command line work as expected?**

Yes

**Does the script parse the sam file correctly?**

**Does it skip the header rows?**

Yes, rows that start with @ are skipped

**Does it extract the correct columns?**

Yes, you extract the sequence id, position, cigar string and NH tag successfully. The position is correctly cast to an integer.

**Is error checking/handling implemented appropriately?**

A try/except block is used when opening the file. You also do some additional checking, for example you verify that the NH tag starts with NH. An additional check might be to use a try/except block when you cast the position to an integer – in a situation where a different sam file is used, this would alert you if the file is not formatted correctly.

**Does the script correctly use the NH tag to find reads that only map once?**

Yes. I would consider converting the numeric value associated with NH tag to an integer, since it is intended to be a numeric value. In this case where you are looking for equality, using strings is ok. However, if you wanted to do a different comparison such as > or < you would need integers.

**Does the script correctly identify intron-spanning reads by looking for the presence of N’s in the CIGAR string?**

Yes

**Does the script correctly infer the locations of the intron junctions by parsing the CIGAR string?**

Yes, your logic for this makes sense and this code works as intended.

**Are intron junctions stored in a suitable data structure to enable counting?**

Yes, you use a dict with an appropriate key comprising the sequence id, start and end for each junction. This enables you to look up each junction as you process it to determine if you have seen it before and to count them.

**Does the script correctly parse the Gene Location file?**

**Is the header row skipped?**

Yes

**Has the necessary data been extracted?**

Yes, you extracted the gene id directly from the file, and you extract the chromosome, start and end from the location string. The start and end are cast to integers.

**Is error checking/handling implemented appropriately?**

Similar to the sam file, you used a try/except block when opening the file. You did not do any additional checks – again using a try/except block when turning the coordinates into integers would have helped you to identify incorrectly formatted input.

**Does the script correctly identify which introns are in which gene?**

Yes, you determine if junctions are in genes by matching the chromosome and by determining that the coordinates for the intron are contained within the gene.

**Is the output table appropriately formatted?**

Yes

**Is the data in the output table correct?**

Yes

**Is the code written in good style?**

**Meaningful variable names?**

Variable names were meaningful throughout and your code was easy to follow.

**Comments?**

Comments were plentiful, and added value to what I could learn from the code.

**Efficiency?**

Your code was reasonably efficient. It did not read files more often than necessary, and it used sensible data structures for fast access where appropriate. One inefficiency is that you are storing the data from the genes file in a dict. This is not strictly necessary – you could cross-reference each gene with your junctions dict as you process them without ever storing them. Your method is inefficient in two ways; firstly, it increases your memory footprint unnecessarily and secondly you are effectively iterating through all the genes twice, once to read them and once to match them with the introns. You could do both these things in a single iteration.

**Use of functions?**

You wrote three functions, one to parse the sam file, one to parse the genes file and one to intersect the two. These functions all worked as expected. However, for the most part, they are far too big and complicated. A good rule of thumb is that one function should do one thing which can be described in a short sentence. Using your process\_sam\_file function as an example, I could describe this by saying that “it parses the file AND it extracts the relevant columns from each line AND it checks the NH tag AND it processes the CIGAR string”. Ideally, I would want to split this sentence where it says “AND” and turn each part into a separate function.

One thing we covered which was missing from your script was using assertions to verify that functions are working. Using smaller functions as described above will help you to do that effectively.

**General Comments**

This was very nice, well done. Everything worked as expected, and the code was easy to follow. The major areas for improvement are highlighted above, but to summarise:

* Try to avoid iterating over the same data more often than necessary – if you can do all of your processing in one pass, you should do that.
* Try to work on creating smaller, simpler functions. This inevitably means you will need to create more of them, but that’s ok. As well as making your code more flexible, this will help you to test your functions effectively.