Breakpoint Analysis

Monday, November 18, 2024 10:27 AM

- PerBaseQualityScores at line 71 a breakpoint was placed and stopped the program. Meaning that getPercentages code is ran.
- Same Module but at line 110 in getOffsets, a breakpoint was placed and it didn't stop the program. After the program was completed it then allowed me to resume.
- Line 131 of PerBaseQualityScores in processSequence a breakpoint was placed. This stopped the program meaning we need this module.
- When running a breakpoint in the PhredEncoding class: uk.ac.babraham.FastQC.Sequence.QualityEncoding.PhredEncoding. It ran as its own main class instead of the whole program. However, when running the main application, the breakpoint didn't stop the program.
- In the QualityCounts class, a breakpoint was placed and it stopped the code from progressing at all with its analysis.

```
private double getPercentile (int minbp, int maxbp, int offset, int percentile) {
        int count = 0;
        double total = 0;

        for (int i=minbp-1;i<maxbp;i++) {
                if (qualityCounts[i].getTotalCount() > 100) {
                        count++;
                        total += qualityCounts[i].getPercentile(offset, percentile);
                }
        }

        if (count > 0) {
                return total/count;
        }
        return Double.NaN;

}
```

This code calls getPercentile() for qualityCounts within a for loop.

Here is the the method it is calling:

```
public double getPercentile (int offset, int percentile) {

        long total = totalCounts;

        total *= percentile;
        total /= 100;

        long count = 0;
        for (int i=offset;i<actualCounts.length;i++) {
                count += actualCounts[i];
                if (count >=total) {
                        return((char)(i-offset));
                }
        }

        return -1;

}
```

This getPercentile is a method of the qualityCounts instance and is essentially a nested for loop.

In PerBaseQualityScores we are looking for heavy math areas that are being called multiple times:

GetMean calculates the mean of the quality counts list. Here is the code within the PerBaseQualityScores Module:

```
private double getMean (int minbp, int maxbp, int offset) {
        int count = 0;
        double total = 0;
```

```
        for (int i=minbp-1;i<maxbp;i++) {
                if (qualityCounts[i].getTotalCount() > 0) {
                        count++;
                        total += qualityCounts[i].getMean(offset);
                }
        }

        if (count > 0) {
                return total/count;
        }
        return 0;

}
```

As you can see it is calling getMean for that method, however there is another nested for loop within that function because getMean is also performing that as well here:

```
public double getMean (int offset) {
         long total = 0;
         long count = 0;

         for (int i=offset;i<actualCounts.length;i++) {
                 total += actualCounts[i]*(i-offset);
                 count += actualCounts[i];
         }

         return ((double)total)/count;
}
```