Name: Michael Rogers
ID: 105667404

CSCI 3104, Algorithms **Profs. Grochow & Layer**
Explain-It-Back 5 **Spring 2019, CU-Boulder**

At a recent infections disease seminar, you hear about how the speaker is sequencing millions of *Plasmodium falciparum* genomes (the human malaria parasite) in order to better characterize how patients respond to treatment. In the presentation, the speaker complained to the audience that although they are making the data sets a small as possible by only using 2 bits to encode the 4 nucleotides of the genomes their IT department is still struggling to store all the data. Later in the presentation, you learn that the Plasmodium falciparum genome is "AT-rich." That is, over 80% of the nucleotides in the genome are either A or T. Please help this team understand how they can leverage Plasmodium falciparum's AT-richness to help their IT department deal with the influx of data.

Name: Michael Rogers
ID: 105667404

CSCI 3104, Algorithms **Profs. Grochow & Layer**
Explain-It-Back 5 **Spring 2019, CU-Boulder**

Dear Plasmodium falciparum team,

It looks like you guys are off to a great start! Bringing down the space taken up by the four nucleotides to 2 bits is already a step in the right direction. I understand that you are still having trouble storing all the data, luckily, I have a strategy that will be a huge help to your IT department. The most important thing we need to utilize is the fact that nearly 80% of the nucleotides are A or T. In computer science a very common and fast way of storing data is something called a hash table. This hash table uses something called a hash function. The hash function acts as an instruction manual for where to store each value in the hash table. Because we have instructions for where to insert and store data, it is very fast to search for and insert data into the table. Unfortunately, if we have a hash function that is specific for each nucleotide, it will still struggle with storing all the data. Therefore, my proposal is this: we use something called a uniformly random hash function. Going through each nucleotide takes too much space to store, however, if we randomize the hash function, there is an 80% chance that one of the nucleotides will be an A or a T. We have now effectively cut down how much space it takes to store these nucleotides, because we don't have to use as much space to store A and T because of our randomized hash function. In hash tables when two sets of data have the same value to be stored, there is something called collision handling. With collision handling these same value data points are stored in a 'bucket' with each other. In a regular hashing function there can be certain buckets that can be overfilled depending on how the function is written. Because our hash function is uniformly random, there is a better chance that all these buckets will be used equally instead of certain ones being used more than others. This helps store the data more efficiently. I do think that this method for storing the genome will help substantially, however, this is a fairly advanced algorithm. If the IT department has any questions for me, inform them they are welcome to come to me with any questions.

Good Luck,
A computer Scientist you know