Quiz Questions

A)Is there a limit on the number of TotalCharges values, which can be added correctly using the library?

Ans:
Yes, there must exist a limit on the number of TotalCharges values, which can be added correctly using the library. As we know that Fully Homomorphic Encryption(FHE) Schemes add noise to the ciphertext and the noise increases with the evaluation of each gate. Though we are using bootstrapping through which noise can be reduced from ciphertext, after a large number of additions(where each addition involves AND,OR and XOR Gate) noise will excceed a threshold which in turn make it impossible to decrypt the ciphertext. Practically it is very difficult to reach such asymptotically large number of operations because time needed to evaluate a single gate operation in FHE is high.

This line from original paper(Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds) supports the answer [In homomorphic encryption, messages are encrypted with a noise that grows at each homomorphic evaluation of an elementary operation. In a somewhat encryption scheme, the number of homomorphic operations is limited, but can be made asymptotically large using bootstrapping.]

B)Running time for addition. Is it possible to parallelize this operation?

Ans:
Assuming addition is done for 32 bit integers(Data present in Total Charges exceeds value 2^16) and with Serial Full Adder circuit(with 2 XOR, 2 AND and 1 OR gate), time taken for adding 10 such numbers take approximately 33 seconds.(on i5 3rd gen processor for libtfhe-spqlios-avx.so, time maybe low for i7 processor with libtfhe-spqlios-fma.so) Hence, for adding 10000 such numbers it will take approximately 9 hours(which is very high number, Hence I was not able to test it).

Parallelization of this operation is possible, though parallelization(using openmp or cuda) is not yet supported completely(Promised in v1.5). Using Look-Ahead Carry Adder parallelization is possible but number of gates required are high due to carry calulation in parallel. Hence it needs more time(Implementation done in second file). Also, with the help of MUX, number of gate operations could have been reduced in carry calculation in parallel, which may reduce running time further.

Without support for complete parallelization and without optimization using MUX, running time of parallel addition(with lookahead carry adder) is approximately 3 Additions per minute which is quite slow. Though it is slow, parallelization of such operations will be useful for performance improvement in FHE.

c) How much does the ciphertext increase in length as compared to the plaintext value?

Ans:
Each ciphertext bit (like ciphertext[i]) takes up 24 Bytes (As per sizeof function). Hence, if plaintext is 32 Bits then ciphertext will take 32*24 = 768 Bytes.
Instead of depending upon what value plaintext stores, it depends on size of the plaintext (16 Bits or 32 Bits).
Storing Ciphertext in a file take approximately 32256 Bytes for 16 bit ciphertext, that is 2016 Bytes for each ciphertext bit.