Introduction

Proteins are one of the most important biomolecules that are involved in almost all biological activities and therefore studying them is a central part of bioinformatics. They consist of a sequence of units called amino acids which fold into a certain 3D shape to do its biological function. Studies on proteins usually involve analyzing their sequence or 3D structure. Obtaining the 3D structure of proteins is an expensive and complicated task and the number of proteins that we have their structure is a small portion of all proteins known to exist. However, having the sequence of proteins is easy and cheap these days and complete set of protein sequences (proteomes) for most of organisms are publicly available in databases like UniProt. Therefore most of genome scale analyses are done on protein sequences and fast operations on sequences are of great importance today. This is especially true because the number of sequences involved in studies are usually from thousands to millions.

Doing analysis and predictions on protein sequences often involves computing some features that captures some property in the sequence. These features then are usually used to perform some predictive modeling or doing some statistical analysis. Computing these features is done independently from each other so there is an inherent potential for parallelism to be utilized by parallel platforms.

Problem Description

A protein sequence is usually in the form of a string of characters each of them representing a specific amino acid. There are 20 amino acids and therefore these strings are made of 20 Letters usually in capital letters. The 20 amonio acids and some of their properties is listed in the table below.

Amino acid	Shown By	Side chain polarity	MW (weight)	Occurrence in proteins (%)
Alanine	A	nonpolar	89.094	8.76
Arginine	R	basic polar	174.203	5.78
Asparagine	N	polar	132.119	3.93
Aspartic acid	D	acidic polar	133.104	5.49
Cysteine	C	nonpolar	121.154	1.38
Glutamic acid	E	acidic polar	147.131	6.32
Glutamine	Q	polar	146.146	3.9
Glycine	G	nonpolar	75.067	7.03
Histidine	H	basic polar	155.156	2.26
Isoleucine	I	nonpolar	131.175	5.49
Leucine	L	nonpolar	131.175	9.68
Lysine	K	basic polar	146.189	5.19
Methionine	M	nonpolar	149.208	2.32
Phenylalanine	F	nonpolar	165.192	3.87
Proline	P	nonpolar	115.132	5.02
Serine	S	polar	105.093	7.14
Threonine	T	polar	119.119	5.53
Tryptophan	W	nonpolar	204.228	1.25
Tyrosine	Y	polar	181.191	2.91
Valine	V	nonpolar	117.148	6.73

(source: wikipedia.org)

To obtain protein-level features, usually the sequence string is explored for different properties. One of the simplest possible features can be the fraction of participation of each amino acid in the sequence of protein. This will result in 20 features one for each amino acid. Another type of features can be that if we want to see if the protein has accumulation of e.g. polar amino acids in some part of its sequence, we can slide a window of certain size over the sequence and take note of the maximum or average number of polar amino acids that we see in the window. Here we focus on three type of features:

- Fraction of one or a group of amino acid participation in the sequence
- The maximum value for a sliding window that counts the number of occurrence of one or a group of amino acids in the window and slides over the sequence from the beginning to the end.
- The maximum value for a sliding window that sums the values of a specific score for one or a group of amino acids in the window and slides over the sequence from the beginning to the end.

Objective

These features are now computed by means of some python scripts that reads and compute their value for these sequences one by one sequentially. It takes about an hour for 200 hundred features and few thousand proteins. Since the computation of these features are completely independent between sequences, these operations can be done in parallel using threads. However, GPU computing can leverage parallelism better than multiple CPU threads because it provides much larger number of threads of execution, given that each of these parallel tasks are well suited for the capabilities of the GPU cores. Based on this idea, in this project, I want to implement parallel computation of these features on CUDA platform hoping to use the computation power of GPUs to reduce the time for the task. This way, more candidate features of the three template types above, with different configuration, can be examined more quickly for correlation with a target variable. This can accelerate the procedure that is recurring frequently in my PhD research.