Project 3: Protein folding

Motivation

Protein folding is an extremely hot topic in medical research these days, unfortunately protein folding is extremely computationally demanding and requires a huge supercomputer to fold even the simplest proteins. Luckily the task of calculating protein foldings is quite well suited for bag-of-task computing.

Proteins are made up of amino-acids, of which there are 20 types. Thus a protein can be viewed as a sequence of amino-acids and folding such a sequence means that the sequence 'curls up' until there is a minimum of unbound energy present in the protein.

Figure 1 Example of a protein fold.

In this class we need not concern ourselves with the chemistry behind protein-foldings. Instead we can play with a simplified version of proteins called prototeins – proto-type proteins.

Prototeins

Our simplified prototeins are folded in only two dimensions and only in 90 degree angles. This is much simpler than real three dimensional foldings with angles depending on the amino-acids that are present at the fold, but as a model it's quite sufficient. Our amino-acids are also reduced to two types; Hydrophobic (H) and Hydrophilic (P). When our prototein is folded it will seek the minimal unbound energy, modeled by the highest number of H-H neighborships.

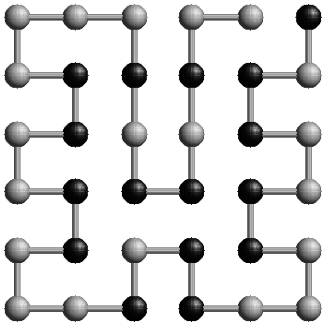

Figure 2 A prototein with 15 H-H bindings.

Even though prototeins seems very simplified we can still learn quite a lot about real protein-foldings from the way the prototeins are folded.

You can read more on prototeins in: http://www.americanscientist.org/template/AssetDetail/assetid/15717.

Programming Task

The solution should be parallelized implemented using PyPastSet, and may be based on the sequential version available. The code should be run on from one through at least 64 CPUs preferably 256, in powers of two.

Report

Your report, which should be submitted through Absalon, should be no longer than 2 pages in total. You should explain how you have transformed the sequential code and how this has influenced your performance. You should provide experiment results with the 3 provided prototeins, where you compare the performance of the original version to your parallel solution. The results should be presented as an easy the read graph, which includes the absolute and relative before and after transformation of the code.

Example performance: