Parallel Feature Extraction By Hidden Markov Model And Parallel K-Mean Clustering For Protein Sequences

Ahmad Tamimi

Supervisor: Mohammed Aldesht

Master of Informatics at Palestine Polytechnic University

Palestine Polytechnic University

Introduction

Hidden Markov models widely used as tool for sequential data modeling, and it were used many times in data clustering. In this work a HMM were employed to build a new space representation as feature extraction for protein sequences. Where each sequence described by a vector of its similarities respect to a predetermined set of other objects. K-mean clustering then used to cluster these set of sequences into K clusters. This work focus in distributing HMM feature extraction process and parallelize the k-means clustering algorithm during the distance calculation and centroids Update phases, based on the master/slave paradigm. This distributed and parallel process is designed in such a way that each P participating node is responsible for handling N/P data sequences.

Project Objectives

1. Find the suitable feature for proteins sequences from different families has a different lengths.
2. Clustering sequences into a known K clusters
3. Built the solution algorithm based on a Parallel approach.
4. Achieving a height clustering accuracy, while obtaining a good speedup results.

Results

1. built a new representation space in which each object is described by the vector of its similarities with respect to a predetermined set of other objects. These similarities are determined using hidden Markov models.
2. Clustering is then performed in such a space using k-mean algorithm.
3. This work algorithm contains two major phases, first one is feature extraction while the second is clustering. This first phase where parallelized by distribute all the sequences to the available process. Let P is number of processor. So each processor should handle N/P sequences. On the other hand parallelized K-means algorithm based on the inherent data-parallelism especially in the Distance Calculation and Centroids Update operations.
4. As table 1 shows, a good accuracy achieved which is by average 98.5%, while a perfect speedup appear, for each new processor added, the speedup increased by one.

Proposed system

This work is looking for obtaining a protein sequence cluster system, that distributes and parallelized its main phases as figure 1 shows, while getting same or better accuracy, with much better performance.

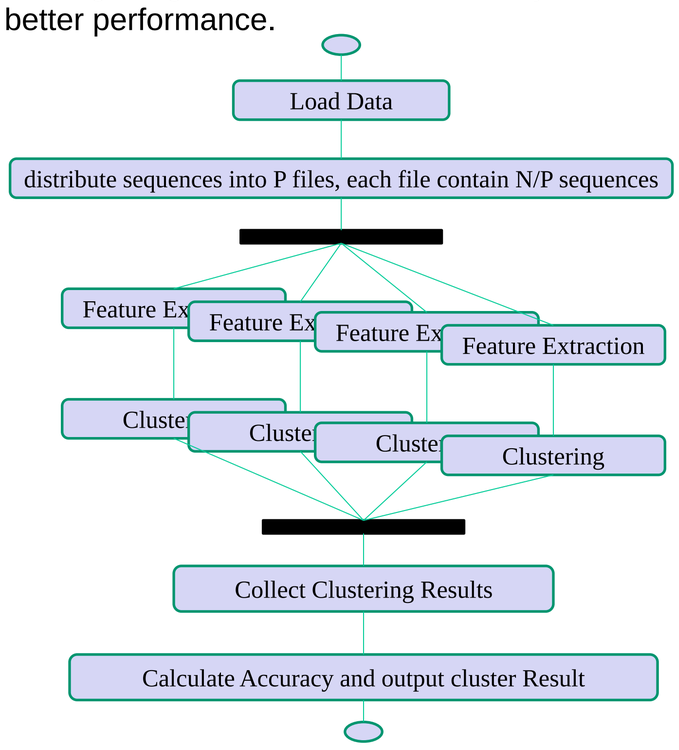

Figure 1: proposed system process

System Design and Implementation

This work method contains two major phases, first one is feature extraction while the second is clustering. Both of them were parallelized. In the following section a detailed description of each phase methodology will be discussed.

Feature Extraction

The method for sequences feature extraction using HMMs can be summarized by the following algorithm. Consider a given a set of N sequences {O1……ON} to be clustered; the algorithm performs the following steps :

1. Train one HMM profile λi for each sequence Oi.
2. Compute the distance matrix D = { D (Oi ; Oj) }, representing a similarity measure between sequences features; this is typically obtained from the forward probability P(Oj| λi) that gained by the HMM profile.
3. The log-likelihood (LL) of each model, given each sequence, that computed in step 2. used to build an LL matrix. This matrix has NxN numeric value. Where each row is a feature for one sequence. That includes the similarity measure from all other sequences.

This first phase where parallelized by distribute all the sequences to the available process. Let P is number of processor. So each processor should handle N/P sequences using the previous algorithm. A good point here is there isn't any communications between nodes during feature extraction.

K-mean Clustering

The most common algorithm uses an iterative refinement technique. This clustering method could be summarized in the following algorithm.

1. Place K points into the space represented by the sequence vectors that are being clustered. These points represent initial group centroids.
2. Assign step: each sequence to the group that has the closest centroid.
3. Update step: recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move or max iteration is reached.

The sequential algorithm spends much of its time calculating new centroids (step 3) and calculating the distances between n data points and k centroids (step 2). execution time can be cut down by parallelizing these two steps

```
Master Algorithm
Load random initial K sequence vectors as centroids
while (true)
    broadcast centroids
    prepare N/P sequence for clustering ( local LL matrix)
    calculate N/P sequences distance from K centroids
    assign each sequence to the nearest centroid
    collect slaves assign result and K size
    calculate new centroids
    if (new clusters = old clusters) or  iteration >= MAX
        kill slaves
        break
    end if
end while
```

Algorithm 1 : Master alg. for k-mean clustering

```
Slave Algorithm
prepare N/P sequence for clustering ( local LL matrix)
while(true)
    receive centroids from master
    calculate N/P sequences distance from K centroids
    assign each sequence to the nearest centroid
    send to master sum of group vector and clusters size
end while
```

Algorithm 2 : Slave alg. for k-mean clustering

experimental results

Number Of Nodes	1	2	4	6	8	10
Avg Comm Time	0	0.14	0.11	0.076	0.15	0.65
Avg Comp Time	5787	1919	1275	1175	1198	1256
Average Idle Time	0	110.8	64.19	56.02	47.23	42.25
Average Real Time	5787	1974	1323	1235	1239	1295
Speedup	1	2.93	4.37	4.68	4.66	4.46
Accuracy (%)	99	98.55	98.58	98.55	98.22	98.45

Table 1: Average exp. results per node number

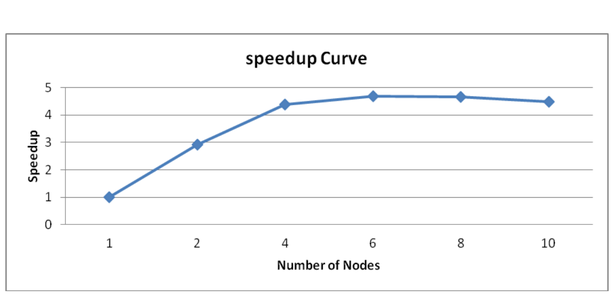

Figure 2 : speedup curves per number of nodes

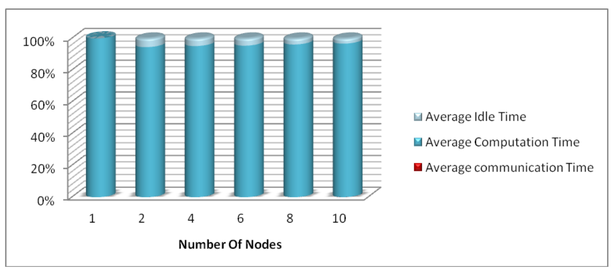

Figure 3 : Execution time distribution