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

AHMAD TAMIMI [1]
MOHAMMED ALDASHT[2]

[1]Master of Informatics at Palestine Polytechnic University
ahtamimi@student.ppu.edu

[2]Assisstant professor at IT department
Palestine Polytechnic University
Hebron, Palestine
mohammed@ppu.edu

ABSTRACT

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.

Keywords: *Hidden Markov Model , K-Mean Clustering, Protien Sequences.*

1. INTRODUCTION

HMM is what used in this study to extract sequence futures, Traditional full connected HMM and profile HMM are the two approaches that used to build a HMM.

Full Connected Hidden Markov Model are statistical models which are generally applicable to time series or linear sequences. They have been widely used in speech recognition applications [1], and have been introduced to bioinformatics in the late 80's [2]. A HMM can be visualized as a finite state machine [3]. Which generalized from a Markov chain, in which each ("internal") state is not directly observable (hence the term hidden) but produces ("emits") an observable random output ("external") state, also called "emission" [4]. In this case, the time evolution of the internal states can be induced only through the sequence of the observed output states [4].

Profile HMMs [5], use the sequence family to build a profile which includes the position-specific probabilities of variation in amino acids, as well as insertions and deletions [3]. This indicates conserved positions which is important to the family, and non-conserved positions which are variable among family members. The sequence, whose structure is not known, is then aligned to the profile, indicating the degree of homology. The membership of a sequence to a family is either given by the most probable path through the model, or by its posterior probability summed over all possible paths [3]. The second approach is what used to build the HMM's, using SAM [7], which is a system software for sequence alignment and modeling that provides the necessary tools for models building and align the test data.

K-means clustering , an unsupervised learning algorithms that have a set of n data points in d-dimensional space Rd and an integer k and the problem

[1] All correspondence should be addressed to the first author.

is to determine a set of k points in Rd, called centers, so as to minimize the mean squared distance from each data point to its nearest center[6].
Unsupervised classification (or clustering) of data is undoubtedly an interesting and challenging research area: it could be defined as the organization of a collection of patterns into groups, based on similarity[8].

2. RELATED WORK

Many researches were released on clustering and feature extracting methods for proteins, in this section we will discuss some of these work and how much it is related to our work.

In [10], F. Othman, and et al, parallelized K-means algorithm based on the inherent data-parallelism especially in the Distance Calculation and Centroids Update operations. each P participating node is responsible for handling N/P data points. each of the P nodes must update and store the mean and k latest centroids in the local cache. The master node will accumulate new assigned data points from each worker node and broadcast new global mean to all. three datasets were used, ribosomal RNA for twenty four organisms, vertebrate mitochondrial DNA sequences and complete genomes of roundworm. They applied the PWM method for feature extraction, where they calculated the frequency of nucleotides A, T, C and G for each position in the sequences.

Othman's work has four differences when compared to this work, first of all Othman work keeps a local copy for the all n data, while this work keep only n/p as local copy at each node, just the master node hold a copy of all data. Secondly Othman calculate a local mean for the centroid before calculating the global in the master node, while this work calculate it once in the master node. Thirdly, he used positional weight matrices (PWM) to extract the feature, while this work use HHM. And finally Othman work using a fixed protein length, while this work has different protein lengths.

In [8],[11], they 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. Clustering is then performed in such a space using many clustering methods the best results comes from k-mean algorithm.

The major difference between [8], [11] and this work is that they use a full connected HMM approach in sequential implementation, while this work used HMM profile. Beside we have a parallelism implementation when they don't.

In [9], Perrone, and et al, described and applied an unsupervised learning K-means clustering algorithm with Hidden Markov Model to address a problem of handwritten character allograph determination.

In [12], A. Britto, and et al, proposed a parallel approach for the K-means Vector Quantization algorithm for the main problem related to VQ which is the training process that requires a large computation time and memory. the two-stage Hidden Markov Model (HMM) based method for recognizing handwritten numeral strings, requires the construction of 3 codebooks. which they built them based on the master/slave paradigm.

The previous tow works didn't use a vector of similarities with respect to a predetermined set of other objects as [8], [11] and this work use, also the domain of them is difference from this one. But the most important thing in his study is the parallel k-mean clustering approach Britto proposed which matches the one we use.

3. METHODOLOGY

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.

1. 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 [8]:

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.

2. K-mean Clustering

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

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. The Distance Calculation operation can be executed asynchronously and in parallel for each sequence feature. The idea is to let each processor in the parallel system P processors to handle the N/p sequences that extract its feature in the previous phase. However, each of the P nodes must assign local sequence to K clusters. then calculates the sum of each group vector values and the size of each cluster. The master node will accumulate new assigned data labels and size of each cluster in K from workers node and broadcast new global mean to all. In this parallelized algorithm the master node has its load too to handle as shown in the following algorithms.

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 algorithms 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 algorithms for k-mean clustering

4. DATASET AND EXPERIMENTAL RESULTS

Dataset that used to test this system contains three proteins families, NUDIX hydrolase (YfcD), Inner membrane transport protein (ydiM), and paraquat-inducible membrane (pqi). Each family contains 100 sequences with different lengths.

Testing Environment that used for collecting results was a virtual machine with four Intel® Core™ i3 CPU @2.13GHz, and 2G ram. The operating system we used is Ubuntu 11.10 with 2G swap memory. And the Lam-MPI were used as parallel tool.

Evaluation of this system comes based on two criteria's, accuracy of clustering results, and the speedup of the parallelism. For each number of nodes parameter we did three experiments. Table 1 shows the average results in details for these experiments.

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 experiments results per node number

All averages time appears in the table were measured in seconds. Average communication, computation, and idle time was calculated per each node number regardless of the real execution time that appear in the fourth row, these values measured based on the slowest node in each experiments.

The speedup results comes by applying Amdahl's Law of Speedup. "Amdahl's law is a model for the relationship between the expected speedup of parallelized implementations of an algorithm relative to the serial algorithm" [Wikipedia]. Which can be expressed in the

following formula, speedup = Time(serial) / Time(Parallel). In this work the speedup obtained by dividing the average real time when having one node – ART(1) – on average real time for parallel nodes – ART(n) –. For example when having two nodes the speedup is ART(1)/ART(2).

For accuracy results we labeled each sequence with its group number before the cluster algorithm started and when it finish we analysis each cluster to calculate out how much each sequence group clustered together.

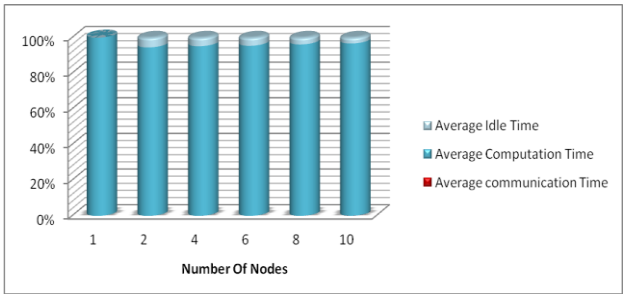

Figure 1 : Execution time distribution

Many information from this table can be obtained. Figure 1 presents the distribution of execution time between communication, computation, and idle time. We note that the communication time is almost zero. This happen because of a small data size that used in the experiments. The idle time is also very small compared to computation time, this happen because our problem contain a big parallel independent portion. This portion is available in feature extraction which take more than 90% of the computation time as figure 2 show.

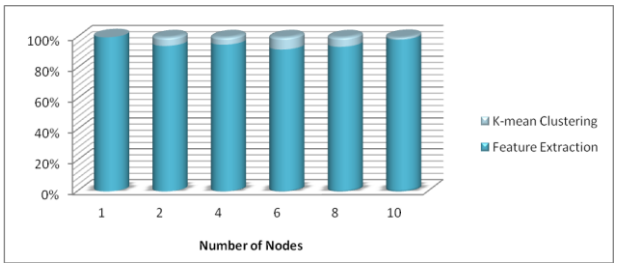

Figure 2 : Computation time ratio for both algorithm phases

Speedup is one of the important criteria's to evaluate this work. A very good result appear as figure 3 shows. The speedup comes equal to number of node until reach 4 nodes, and this increasing will continue if this experiments tested in real nodes rather that four cores in a virtual machine. This vision comes from the time distribution results that appeared in figure 1, which shows that there is a very low coupling attribute between data process that reflects in a low communication and high computation times. Also figure 2 shows that the ration of Feature Extraction phase to K-mean Clustering phase is too big knowing that there isn't any communication between nodes in this phase case this perfect utilization.

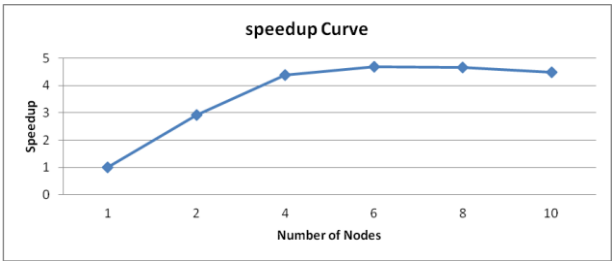

Figure 3 : speedup curves per number of nodes

Accuracy is another important criteria to measure and traced. A grate results appear in figure 4. This results comes because of the strength of HMM mathematical modeling, and the new feature space. Another thing to report is the small changing in the accuracy when number of parallel node changes

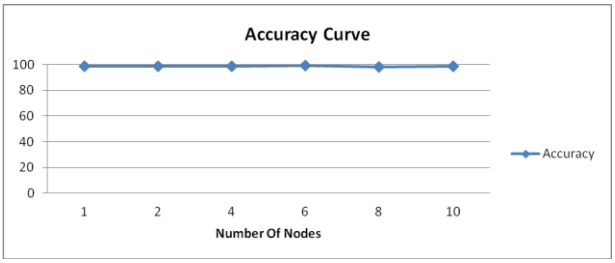

Figure 4 : accuracy curve per node number

5. CONCLUSION

The By parallelizing both system phases, the feature extraction phase that using HMM to build a new space representation -as feature extraction for protein sequences-. and the K-mean clustering phase that used N nodes to cluster these set of sequences into K clusters. based on the master/slave paradigm. We achieved a very good result in both major measurement criteria accuracy and speedup. The speedup we achieve is equal to number of node until four nodes, and obtain 98.5% as clustering accuracy.

A lot of work can be done via this algorithm approach. It seems that this work will give much better results if it executed on real nodes cluster rather than virtual machine. Also it is clear the k-mean clustering will give

a negative result for small data, since it efficiently will appears when having a huge amount of data.

REFERENCES

[1] L. R. Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition," IEEE, 1989.

[2] G. A. Churchill. "Stochastic models for heterogeneous DNA sequences," Bull Math Biol, 1989.

[3] N. Mimouni, G. Lunter, and C. Deane. "Hidden Markov Models for Protein Sequence Alignment," University of Oxford, 2004

[4] V. Fonzo, F. Aluffi-Pentini, and V. Parisi. "Hidden Markov Models in Bioinformatics," Bioinformatics, 2007, 2, 49-61, 2007

[5] D. Haussler, A. Krogh, I. S. Mian, and K. Sjölander. "Protein Modeling using hidden Markov models: Analysis of Globin," Proceedings of the Hawaii International Conference on System Sciences, 1, IEEE Computer Society Press, 1993

[6] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silvernmane, A. Wu."Efficient k-Means Clustering Algorithm, Analysis and Implementation," IEEE, 2002.

[7] R. Hughey, K. Karplus, and A. Krogh. "Sequence Alignment and Modeling Software System," University of California , 2003.

[8] M. Bicego, V. Murino, and M. A.T. Figueiredo. "Similarity-Based Clustering of Sequences using Hidden Markov Models," University of Verona, Italy, 2002.

[9] M. P. Perrone, and S. D. Connell. "K-Mean Clustering for Hidden Markov Model," Pen Technologies Group, IBM, Waston Research center, 1999.

[10] F. Othman, R. Abdullah, N. Addul Rashid, and R. Abdul Salam. "Parallel K-Means Clustering Algorithm on DNA Dataset," University of Sains Malaysia, Malaysia, 2003.

[11] P. Smyth. "Clustering sequence with HMM," university of California,1997.

[12] A. Britto , P. Souza, R. Sabourin, S. Souza, and D. Borges. "A Low-Cost Parallel K-Means VQ Algorithm Using Cluster Computing," In Proceedings of ICDAR, 2003.