Comparing two Algorithms for Clustering Aligned Pattern Clusters

Sanderz Fung
Systems Design Engineering
University of Waterloo
Waterloo, Canada
s3fung@uwaterloo.ca

En-Shiun Annie Lee
Systems Design Engineering
University of Waterloo
Waterloo, Canada
annie.lee@uwaterloo.ca

Andrew K.C. Wong
Systems Design Engineering
University of Waterloo
Waterloo, Canada
akcwong@uwaterloo.ca

ABSTRACT

Advances in high-throughput bioinformatics have provided a large influx of novel sequences, thus making the analysis of the sequences crucial. Proteins sequences are composed of amino acid alphabets; network clusters of protein regions can be analyzed to reveal inherent biological knowledge. The important protein regions are represented by frequent sequence patterns in protein families, which we represent with Aligned Pattern Clusters (APCs). When two conserved protein regions occur simultaneous in one protein, this implies that they interact within the protein. This co-occurrences is used to cluster APCs into APC Clusters. The purpose of this paper is to compare two clustering algorithms for finding these APC Clusters: 1) maximum spanning tree with minimal cut and 2) k-means clustering. We compare the two clustering algorithms by performing three sets of experiments: synthetic dataset, two biological case studies, and a large-scale biological study. The biological results are further confirmed by their correspondence to the three-dimensional structure of the protein. We conclude that k-means clustering performs better than MST with minimal cut due to faster runtime, global optimization and consistence of the final results. Our method and results are currently being verified by important proteins crystallized from an experimental lab in immunology.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Relevance feedback*; J.3 [**Life and Medical Sciences**]: Biology and Genetics

General Terms

Measurement, Experimentation

Keywords

Sequence, Clustering, Ubiquitin, Triosephosphate isomerase, Co-occurrence, Pattern, MST Clustering, K-means clustering

1. INTRODUCTION

Proteins are involved in complex biological process responsible for life. One specific protein sequence is comprised of amino acids from a twenty letter alphabet. The linear sequence of amino acids folds up into a three-dimensional structure. A family of proteins all have the same function, which implies similar three-dimensional structure; however, the protein structure is the same while the protein's sequences may vary. We assume that protein regions of importance within the protein will maintain their conserved segments (i.e. sequence patterns). In this manner, two protein regions of importance that function together will have conserved patterns in the co-evolution process.

Finding protein regions of importance will help identify structural and functional relationships within the protein. There are several approaches for identifying protein regions: database searches and motif-finding. Several databases store the pattern confirmed by experts. In particular, PRINTS [1] provides protein fingerprints found in protein families. Because PRINTS results are comparable to ours, we were able to use PRINTS to verify our patterns. However, some protein families studied, like triosephosphate isomerase, were not stored in the PRINTS database. Other methods incorporate additional biological information to motif finding for finding protein regions. For example, Kinjo and Nakamura [7] finds protein function by comparing patterns of the three-dimensional structure, and Francisco et. al [6] find DNA binding sites using motifs and clustering their co-occurrences. However, these methods use additional information that is difficult to acquire without additional weblab technology. We use sequence data only. To confirm our capability, we use existing protein family data acquired from pFam.

To examine the relationship between two or more protein regions, we find APC clusters using two possible clustering algorithms. For comparison, we devise a method with three parts. The first two parts of our method is based on our previous works [12, 8], to discover, cluster and align similar sequence patterns into Aligned Pattern Clusters, APC [8]. The third part is to determine highly co-occurring APCs on the same proteins using two different clustering algorithms: maximum spanning tree with minimal cut, and k-means clustering. The biological results the clustering are important for interpreting biological knowledge and are confirmed computationally by the protein's three-dimensional structure. Furthermore, they are currently being verified through experiments on an biological application in immunology.

2. METHOD

Our methodology combines three algorithms: the first two from our existing published work and the final algorithm is the main focus of this paper. First, we use a pattern discovery algorithm [12] to discover and locate significant sequence patterns from a protein family while pruning the redundant ones. Next, we apply an APC Algorithm [8] to obtain a list of condensed APCs with variations. Finally, we cluster the discovered the APCs into APC clusters using two different types of clustering algorithms (Fig. 1).

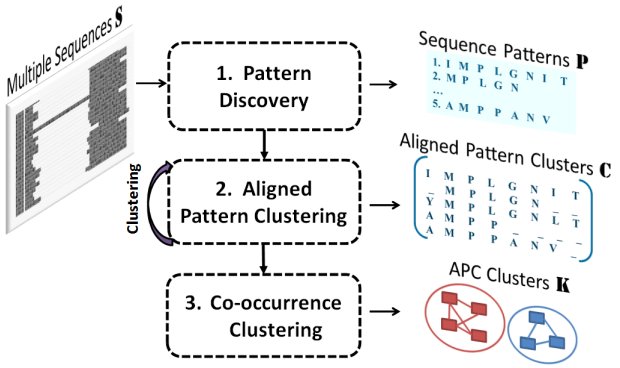

Figure 1: The overall process of our methodology is a combination of three algorithm: 1) the pattern discovery algorithm, 2) the APC algorithm, and 3) the APC clustering algorithm.

2.1 Pattern Discovery

First we discover sequence patterns from the family of protein sequences using a previously developed pattern discovery algorithm [12]. Sequence patterns are statistically significant amino acid association defined as an interdependent ordered sequence of symbols $p = s^1 s^2 ... s^n$ from the alphabet Σ. The pattern p has length n, and the i^{th} symbol that appears in the sequence is s^i. The list of patterns resulting from the pattern discovery algorithm are $\mathbb{P} = \{p^i | i = 1, ..., |\mathbb{P}|\} = \{p^1, p^2, \ldots, p^{|\mathbb{P}|-1}, p^{|\mathbb{P}|}\}$. This resulting list of patterns is pruned of redundant patterns.

2.2 Aligned Pattern Clustering

An APC describes a set of aligned similar sequence patterns with variations. Algorithmically as defined in [8], it is a set of patterns where gaps and wildcards are added to the patterns in order to maximize the similarity between the patterns. Let a set of APC be defined as [8],

$$\mathbb{C} = \{C^l | l = 1, ..., |\mathbb{C}|\} = \{C^1, C^2, \ldots, C^{|\mathbb{C}|-1}, C^{|\mathbb{C}|}\}$$

and let an APC be defined as,

$$C^l = \texttt{ALIGN}(\mathbb{P}^l), \quad (1)$$

$$= \begin{pmatrix} s_1^1 & s_2^1 & \ldots & s_n^1 \\ s_1^2 & s_2^2 & \ldots & s_n^2 \\ \vdots & \vdots & \vdots & \vdots \\ s_1^m & s_2^m & \ldots & s_n^m \end{pmatrix}_{m \times n} = \begin{pmatrix} p^1 \\ p^2 \\ \vdots \\ p^m \end{pmatrix}, \quad (2)$$

$$= \begin{pmatrix} p^1 & p^2 & \ldots & p^m \end{pmatrix}. \quad (3)$$

where $s_j^i \in \Sigma \cup \{-\} \cup \{*\}$ is an pattern p^i with a newly column index j. Each of the $|\mathbb{P}^l| = m$ patterns in the rows of C^l is of length $|C^l| = n$.

2.3 APC Clustering

Co-existence of patterns in different locations of the same protein may indicate that they are functionally related and important for the protein family. In APC clustering, we first find the co-occurrence between APCs using a co-occurrence score. Treating the co-occurrence score between APCs as a similarity measure, we apply clustering algorithm to obtain clusters of APCs with high occurrence score. We consider two clustering algorithms: the first is a graph algorithm that discovers the maximum spanning tree and cut the minimal edge; the second is the k-means clustering algorithm. In each clustering algorithm, we find the optimal number of APC clusters using five clustering indicators. After finding the APC clusters, we confirm the results in the three-dimensional structure by taking the APC clusters with high co-occurrence score and compare the locations of their APCs in the aligned protein sequence with the corresponding three-dimensional structure

2.3.1 Co-occurrence Score

First, we try to find the APCs with the highest co-occurrence on same protein sequences. We first compare all possible APC pairs, using a co-occurrence score to evaluate the similarity between them. The co-occurrence scores quantifies how often two APCs appear together on the same sequence. To calculate the co-occurrence score, Jaccard index is adopted [9]:

$$J = \frac{|C^1_{seq} \cap C^2_{seq}|}{|C^1_{seq} \cup C^2_{seq}|}$$

where

C^1_{seq} = sequences that contain patterns from APC C^1

C^2_{seq} = sequences that contain patterns from APC C^2

The APC pairs are ranked in the descending order of their co-occurrence scores. When two or more APC pairs have the same co-occurrence score, the size union of the two APCs ($|C^1_{seq} \cup C^2_{seq}|$) is used as a secondary ranking criteria. An APC pair with a higher union size indicates that it covers more sequences, and hence, should be ranked higher. A special rule that prevents clustering any APC pair that overlaps each other in any sequence. For example, assume C^1 is "CAT" and C^2 is "ATM". If in any of the sequences, the pattern "CATM", appears, our algorithm would automatically return a co-occurrence score of 0 because of the overlap of AT by the two APCs, despite the fact that there might be other sequences where the two APCs appear together without overlapping. This special rule was applied because overlapping showed that two APCs are similar in both composition and location, and should be joined together as one APC.

Jaccard index was compared to Sørensen coefficient [11] and Mountford's index of similarity [11]. As discussed in [11], Mountford's is a flawed index, and finding that Jaccard in-

dex had the same rankings as Sørensen's and not Mountford's convinced us that Jaccard index is a good fit as the co-occurrence score.

2.3.2 Clustering Algorithms

Next, a set of closely related APCs called APC clusters is found using co-occurrence scores as the similarity measure (or a distance measure in reverse) between two APCs. Two clustering algorithms are considered, maximum spanning tree with minimal cut, and k-means clustering.

Maximum Spanning Tree with Minimal Cut. For the first clustering algorithm, a graph algorithm is considered. Let $G = (V, E)$ be a relationship graph of the APCs. Let each vertex v be an APC and let each edge e be the relationship between two APCs, in this case the co-occurrence score between two APC vertices. A maximum spanning tree (MST) is built using Prim's algorithm and cut the minimal edge of the MST to separate the vertices, which are APCs, into two co-occurrence clusters.

Algorithm 1 Maximum Spanning Tree (based on Prim's Algorithm)

Input: A set of APCs C as vertices V, and the co-occurrence scores between all pairs of APCs as edges E
Output: A set of $|C| = |V|$ edges for the maximum spanning tree edges $E_M = \{e_1, e_2, ..., e_{|V|-|components|}\}$
repeat
  Add any edges e that connects to v to edge list, making sure that the other vertices connected to that edge is not already seen
  if edge list is not empty, **then**
    get the maximum edge from list
    Let the vertex that is connected by the maximum edge but currently not in MST be the new v
    Add the maximum edge to MST edge list
    Add the vertex to the seen vertices list
  else if edge list is empty **then**
    Find a random vertex that is not seen yet in the seen vertices list to be the new vertex
  end if
until all vertices are seen

Algorithm 2 Automatically Optimize Minimal Cut

Input: A set of APCs C as vertices V, and the maximum spanning tree edges E_M
Output: APC clusters $K_1...K_k$, where each K is an APC cluster that contains a set of APCs C
repeat
  Compute cluster indicators for all MST edges
  Sort the cluster indicators
  Select the edge with the optimal cluster indicator
  Cut the edge if the current cluster indicator is better than the previous cluster indicator
until optimal clustering is reached: current cluster indicator is worst than the previous cluster indicator

K-Means Clustering. For the second clustering algorithm, k-means clustering algorithm is modified [9] to cluster APCs. The algorithm uses centroids, a central point that represents a cluster. Firstly, APCs are used as centroids, since calculating a centroid with only co-occurrence between APC is difficult. Secondly, the algorithm is modified to prevent an APC from being clustered with a centroid to which it is not connected [10]. For example, it is possible for APCs not to be connected if they do not co-occur on any sequences, thus causing these two APCs to be in separate APC cluster, where each cluster is fully connected but is isolated from the other. Without additional variables others than the cluster size, only one solution is found for each cluster size. This is different from MST with minimal cut, as the latter depends on clustering indicators also.

Algorithm 3 Modified k-means clustering

Input: A set of APCs C, and the co-occurrence scores between all pairs of APCs J, final number of clusters the k-means clustering is k
Output: APC clusters $K_1...K_k$
Initialize centroids $M_1...M_k$, where each M_1 represent the center of APC cluster K_i
Find number of components
Select first APC from each component as the centroid
for $i = |components| + 1$ to k **do**
  Identify the APC that forms the lowest co-occurrence score with known centroids
  Assign this APC as a new centroid
end for
repeat
  for all $APCC \in \mathbb{C}$ **do**
    Assign C to closest centroid M_j such that C and M_j are from the same component
  end for
  for all $clusterK_i \in \{K_1...K_n\}$ **do**
    Update centroid M_i by selecting APC that maximizes co-occurrence within all APCs in K_i
  end for
until convergence
return $\{K_1...K_k\}$

2.3.3 Clustering Indicators

Finally, to ensure clustering provides the best possible results, five clustering indicators were computed to determine the optimal final number of clusters to be adopted for the APC clustering. All five cluster indicators follow the principle that the average co-occurrence score within a cluster should be maximized and the average co-occurrence score between clusters should be minimized. In addition, additive smoothing was applied to several indicators to prevent division by zero thus causing an indicators values from becoming infinity. The variables and indicators are defined as follows:

k =number of clusters

$s(K_i)$ =average co-occurrence score in cluster i

$s(K_i, K_j)$ =average co-occurrence score between cluster i and j

Average Score

$$\frac{\sum_{i=1}^{k} s(K_i)}{k}$$

Intra / Inter

$$\frac{k + \sum_{i=1}^{k} s(K_i)}{k + \sum_{x=1}^{k} \sum_{y=x+11}^{k} s(K_x, K_y)}$$

Intra - Inter

$$\sum_{i=1}^{k} s(K_i) - \sum_{x=1}^{k} \sum_{y=x+11}^{k} s(K_x, K_y)$$

Dunn index [4]

$$\frac{2 - \max_{1 \le x,y \le k : x \ne y} s(K_x, K_y)}{2 - \min_{1 \le i \le k} s(K_i)}$$

Davies-Bouldin index [3]

$$\frac{1}{k} \sum_{i=1}^{k} \max_{1 \le x,y \le k : x \ne y} \frac{4 - s(K_i) - s(K_j)}{3 - s(K_i, K_j)}$$

For both the Dunn and Davies-Bouldin indexes, the difference of the co-occurrence score $(1 - s(K_i))$ was taken as the distance between two clusters, as required by the indexes' definition. In order to find the optimal cluster count, the minimum of the Davies-Bouldin index and the maximum of the other four indicators was computed, and selected.

2.3.4 *Runtime Comparison*

MST with minimal cut takes $O(n^4)$ to find the optimal cluster count. The maximum spanning tree algorithm takes $O(n^2)$ (Algorithm 1), as it is based on Prim's Algorithm. Afterwards, in Algorithm 2, we take $O(n^2)$ to calculate clustering indicator for each MST edge. Since we need to calculate the clustering indicator for all MST edges, which would have a maximum of $n-1$ edges at each iteration. Computing cluster indicators for all MST edges would take $O(n^3)$. Finally, since for each of the outer loop one edge is removed, there is a maximum of $n-1$ iterations. Hence, finding the optimal cluster count for MST with minimal cut takes $O(n^4)$.

K-means clustering algorithm takes $O(n^3)$ to find the optimal cluster count. Clustering using k-means clustering with a given cluster count k takes $O(n^2)$. For each iterations, k-means clustering first takes $O(kn)$ to compare each APC to each centroid, then takes $O(n^2)$ (if there exist a cluster that includes most of the APCs) compare each APC to all other APCs in the same cluster to update centroids. Since $k \le n$, each iteration takes $O(n^2)$. K-means algorithms will quickly converge, hence the necessary iteration tends to be small can be considered to be linear [9]. Hence k-means clusters will take $O(n^2)$. However, similar to Algorithm 2, we want to find to find the optimal cluster count, and hence will run k-means clustering a maximum of n times, with each run also calculating the cluster indicators, which takes $O(n^2)$. Hence, the whole algorithm takes $O(n^3)$ in the worst-case scenario.

Figure 2: Section of synthetic data 2, showing the character sequences and the two APCs obtained from the sequences.

Therefore, k-means clustering has better runtime of $O(n^3)$ than MST with minimal cut, $O(n^4)$, for finding the optimal cluster count. In addition, k-means just needs to run once, $O(n^2)$, if given the cluster count. This is better than MST with minimal cut, which still needs to calculate clustering indicators for each MST edge before cutting. That takes $O(n^3)$. Hence, unless MST with minimal cut provides better results, k-means clustering should be used instead, due to its better running time.

3. RESULTS

To compare the two clustering algorithms, MST with minimal cut and K-Means clustering, we applied the algorithms to 3 synthetic dataset, and to protein sequences obtained from Pfam [5]: triosephosphate isomerase, ubiquitin and other proteins. Each dataset consists of a list of character sequences. Figure 2 displays a section of second synthetic dataset. For biological datasets, each character represent an amino acid, each sequence represent a protein and all sequences in the dataset are from the same protein family. We also verified the results from biological dataset with Protein Data Bank (PDB) [2] structure, observing any characteristics shared between the APCs found, especially the structural distance between the APCs.

3.1 Synthetic Dataset

Three synthetic dataset were created to evaluate the two APC clustering algorithms before applying them to biological data. The first two datasets have clean APC clusters and the last dataset have less clean APC clusters.

The first dataset consists of five APCs, constructed so that two clean APC clusters are formed after applying APC clustering. The two APC clusters are as follows: 1) three APCs with a co-occurrence of 1 for all pairs 2) two APCs with co-occurrence of 1. There are no co-occurrence between the APC pairs no of the same cluster. For the first dataset, both APC clustering algorithms indicated that the two clusters configuration is optimal, as intended. Also, as shown in Figure 3, both APC clustering algorithms had the same value for each clustering indicators, showing that both clustering algorithm provided the same solution to this dataset.

Building on the first dataset, the second synthetic dataset added an additional cluster and created co-occurrence between APC pairs not of the same APC cluster. In particular, one of the APC in an APC cluster that has a co-occurrence score with an APC not of the same APC cluster (with score just under 0.5). The intended solution is three clusters, but the addition of the co-occurrence edge mentioned above makes it harder for the clustering algorithm to find the correct answer. While k-means was able to find the correct optimal cluster count for all five clustering indi-

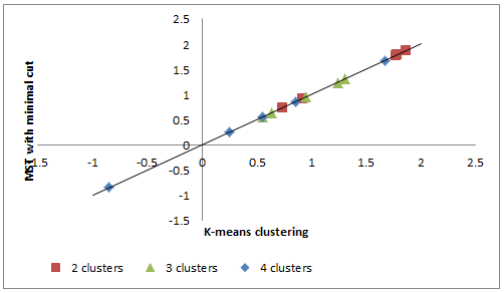

Figure 3: Scatter plot of the clustering indicator values between the two co-occurrence cluster algorithms on synthetic data 1. The line represent the when the two APC clustering algorithms has the same value, which every points lies on.

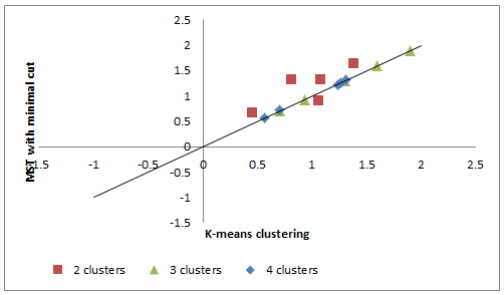

Figure 4: Scatter plot of the clustering indicator values between the two cluster algorithms on synthetic data 2. The line represent the when the two APC clustering algorithms has the same value. The plot shows a positive relationship between the results of the two algorithms.

cators, two of five clustering indicators failed to obtain an optimal cluster count of three for MST with minimal cut. This example showed that k-means clustering performs better at finding the optimal cluster count. Figure 4 plots the clustering indicators scores between the two clustering algorithms under different cluster counts. The figures show that while there are differences, or deviations, between the two clustering indicators, the overall relationship between the results of the two algorithms are still quite consistent.

The third synthetic dataset consists of 7 APCs, with each having similar co-occurrence score with two other APCs, forming a circle-like shape between the APCs, hence making clustering the APCs difficult. The purpose of this dataset is not to test the correctness of the algorithms but rather to test them in unknown or unclear situations. By a majority of 3-to-2 clustering indicators, k-means calculated that the three is the optimal cluster count; MST with minimal cut calculated two as the optimal count by a majority of 3-to-2 clustering indicators. The difference in results shows that given a less clear dataset, the two algorithms will deviate in results. And unlike the first dataset, MST with minimal cut will start providing different answers depending on the clustering indicator used even if the cluster count is the same. However, despite the differences in results, both algorithms were able to provide the same results, highly con-

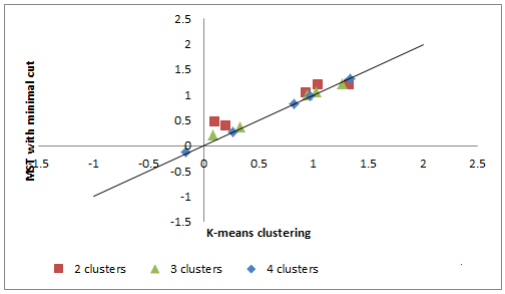

Figure 5: Scatter plot of the clustering indicator values between the two cluster algorithms on synthetic data 3. The plot shows a positive relationship between the results of the two algorithms.

Table 1: APCs discovered in triosephosphate isomerase protein sequences from Pfam.

APC ID	Sequence
C^1	IAYEPVWAIGTG
C^2	IGHSERR
C^3	ILYGGSV
C^4	WAIGTGK
C^5	LVGGASL
C^6	GNWKM

nected APC clusters, the cluster that has the highest average co-occurrence score between its members. Figure 5 plots the clustering indicators scores between the two clustering, which shows the linear relationship, indicating closely similar values in the clustering indicator variables between the two algorithm's results despite the differences between the two algorithms.

3.2 Biological Case Studies

Triosephosphate isomerase and ubiquitin are used as biological protein datasets to compare and analyze the results of both APC clustering algorithms.

3.2.1 Triosephosphate isomerase

With triosephosphate isomerase, we show that k-means clustering is again more preferable than MST with minimal cut. Table 1 lists the six APCs obtained from pattern discovery and pattern alignment on triosephosphate isomerase protein sequences.

Using co-occurrence scores of APC pairs as the similarity measure between APCs, both MST with minimal cut and k-means clustering algorithm were used to cluster the APCs. Each of the the five clustering indicators were used separately to obtain the optimal number of clusters (Table 2). According to results tabulated in Table 2, all five clustering indicators lead to the same optimal cluster number of two. However, the composition of the clusters returned from K-means and MST algorithms are obtained by these algorithms are ($\{C^1,C^2,C^3\}$, $\{C^4,C^5,C^6\}$) and ($\{C^1,C^2,C^5\}$, $\{C^3,C^4,C^6\}$) respectively. They are different. In addition, k-means outperformed for all clustering indicators aside from Dunn index for cluster count of two. Hence we, decide to use the highly connected APC cluster from k-means clustering

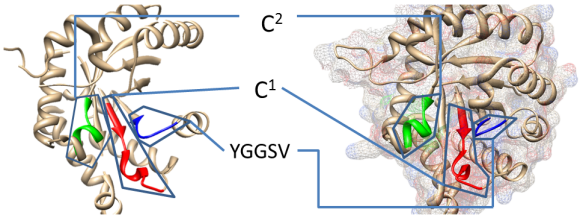

Figure 6: Three-dimensional structure of a triosephosphate isomerase protein found in E. coli (Escherichia coli, PDB ID 4iot). YGGSV is a subpattern of C^3. Inspecting both the spatial distance and the structure, the three selected APCs are close together.

Table 3: APCs discovered in ubiquitin protein sequences from Pfam.

APC ID	Sequence
C^1	DQQ[RK]LI[FY][AS]GKQLED
C^2	ESTLHLVLRL
C^3	DYNIQKE
C^4	VKAKIQ
C^5	KTLTGK
C^6	KEGIP
C^7	SSDTI
C^8	VLRLR

$\{C^1, C^2, C^3\}$ for verification.

A three-dimensional structure of triosephosphate isomerase (PDB ID 4oit) was used to verify the APC pair (Figure 6). Not only did the APCs appear in the structure, the APCs are close to one another in spatial distance, all less than the average distance of 22.35 Å(11.63 Å, 14.22 Åand 7.37 Å).

3.2.2 Ubiquitin

Using ubiquitin to compare the two clustering algorithms provides another example to show that k-mean is superior to MST. Table 3 lists the APCs obtained from pattern discovery and pattern alignment on ubiquitin protein sequences from Pfam. Using co-occurrence score as similarity measure between APCs, we obtained the optimal APC clustering results as tabulated in Table 4.

Similar to results obtained from the triosephosphate isomerase, both clustering algorithm rendered the same cluster count but they provided different cluster composition. In particular, MST with minimal cut provided two different cluster components for different cluster indicators, with clustering indicators Average Score and Intra-Inter returning a different cluster component than the other three indicators. In addition, shown in Table 4, those other the three clustering indicators used for MST with minimal cut returned the same results as k-means clustering. Since eight out of ten clustering indicators agree that the optimal answer is two cluster with the results from k-means, we will use the highly connected cluster $\{C^7, C^4\}$ to verify the three-dimensional structure of ubiquitin (PDB ID 1ubq) in Figure 7. Similar to triosephosphate isomerase, the distance between the two APCs (10.74 Å) is less than the average distance of 14.83 Å.

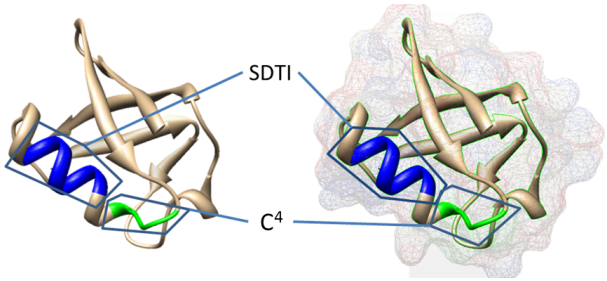

Figure 7: Three-dimensional structure of an ubiquitin protein found in humans (Homo Sapiens, PDB ID 1ubq), with the 5 APCs highlighted. SDTI is a subpattern of C^7. The two APCs are both close in three-dimensional and sequential distance.

3.3 Overall Biological Results

Finally, in order to provide further support that our method using k-means algorithm is produce more accurate results, we further compared these two clustering algorithms for several other protein families.

The results from applying k-means and the MST clustering algorithm to the protein families is given in Table 5 and Table 6 respectively. The final optimal cluster number is the statistical mode of optimal cluster numbers from the 5 clustering indicators. Similar to ubiquitin, while we were able to find a mode for the optimal cluster count, there were still variation in the clustering results given the optimal cluster. Due to the consistency of the optimal number of clusters in k-means, we use the results from k-means for three-dimensional structure verification. Surprisingly, some of the results from MST with minimal cut also share the same highly connected clusters determined from k-means clustering, adding support to the highly connected clusters from k-means clustering are important in the protein family. Both highly connected clusters found through k-means clustering for each protein family and the similarity of the clusters between the two clustering algorithm is in Table 7.

4. DISCUSSION OF ALGORITHM COMPARISON

4.1 Synthetic Results Comparison

The results from synthetic datasets shows that while there are differences between the results from both clustering algorithms, they give a positively linearly correlated result. The results from the first synthetic data shows an exact match of the clustering indicators between the two algorithms. This means that their cluster configuration is exactly the same. However, for the second synthetic dataset, some of the results for MST with minimal cut indicated that the optimal cluster count was two, different from the k-means' and expected solution of three. While by majority, MST with minimal cut still indicated that the optimal cluster count was three, a few still gave the incorrect optimal cluster count in the simple dataset. The results from the second dataset showed that k-means clustering algorithm is more accurate in finding the optimal cluster count.

However, despite differences in results between MST with minimal cut and k-means clustering, both algorithms re-

Table 2: Bolded number indicates the optimal cluster number according to each clustering indicator to be used for clustering on triosephosphate isomerase

Number of Clusters	Average Score	Intra / Inter	Intra - Inter	Davies-Bouldin index	Dunn index
K-means					
2	**0.64**	**1.28**	**0.73**	**1.11**	**0.97**
3	0.45	0.95	-0.25	1.34	0.69
4	0.19	0.65	-2.56	1.58	0.65
MST with Minimal Cut					
2	**0.62**	**1.26**	**0.67**	**1.14**	**1.00**
3	0.48	0.95	-0.24	1.32	0.70
4	0.39	0.76	-1.73	1.45	0.65

Table 4: Bolded number indicates the optimal cluster number according to each clustering indicator to be used for clustering on ubiquitin

Number of Clusters	Average Score	Intra / Inter	Intra - Inter	Davies-Bouldin index	Dunn index
K-means Clustering					
2	**0.35**	**1.21**	**0.46**	**1.19**	**1.05**
3	0.24	1.03	0.10	1.30	0.86
4	0.34	0.98	-0.13	1.27	0.82
MST with Minimal Cut					
2	**0.41**	**1.21**	**0.48**	**1.19**	**1.05**
3	0.42	1.10	0.39	1.19	1.04
4	0.41	0.98	-0.13	1.24	0.92

Table 5: Optimal cluster count for k-means clustering on each protein family.

Pfam ID	Average Score		Intra / Inter		Intra - Inter		Davies-Bouldin index		Dunn index		Final Size
	Size	Score	Size	Score	Size	Score	Size	Score	Size	Score	
PF00061	6	0.30	2	1.21	5	0.85	2	1.19	2	1.07	2
PF00556	6	0.26	3	1.08	6	0.39	6	1.31	3	0.90	6
PF00033	5	0.23	2	1.07	5	0.23	5	1.26	2	0.97	5
PF00115	3	0.31	3	1.21	3	0.69	3	1.21	3	1.01	3
PF00124	5	0.65	2	1.44	2	0.98	2	1.01	2	1.23	2

Table 6: Optimal cluster count for MST with minimal cut on each protein family.

Pfam ID	Average Score		Intra / Inter		Intra - Inter		Davies-Bouldin index		Dunn index		Final Size
	Size	Score	Size	Score	Size	Score	Size	Score	Size	Score	
PF00061	7	0.30	3	1.20	5	0.95	4	1.19	4	1.04	4
PF00556	7	0.29	4	1.16	9	1.06	3	1.25	4	1.04	4
PF00033	3	0.41	2	1.19	4	0.67	2	1.21	3	1.04	2,3
PF00115	2	0.42	2	1.33	4	0.90	2	1.11	2	1.09	2
PF00124	4	0.75	2	0.68	3	1.35	2	0.95	2	1.23	2

Table 7: Details for the APC cluster used to verify on a PDB structure and its similarity to APC cluster from MST with minimal cut.

Pfam ID	Max Cluster average	Max Cluster in k-means	PDB ID	# of MST results	same APC cluster?
PF00061	0.30	2	4GNY	2	Yes
PF00556	0.46	2	1NKZ	1	No
PF00033	0.48	6	1Q90	3	No (subset only)
PF00115	0.52	4	3O0R	2	Yes
PF00124	0.64	4	2GNU	2	Yes

turned the same highly connected cluster. This observation exemplified the differences in the two algorithms: k-means clustering clusters all APCs and MST with minimal cut finds a good cluster.

4.2 Biological Results Comparison

Similar to the third set of synthetic data, the more complex biological datasets reflect the same level of differences between the results obtained from the two clustering algorithms. For triosephosphate isomerase, both algorithms had the optimal cluster count of three; however, they each provided a different highly connected APC cluster. We discovered that MST with minimal cut would have to cut a non-MST edge in order to result in highly connected APC cluster to obtain the same result as that of k-means. The different resulting APC cluster demonstrated that cutting MST edges limited the algorithm to a smaller set of results. Once again, this also implies that k-means algorithm produce more accurate results for the given datasets.

For all other biological datasets used, we saw that MST with minimal cut provided more than one set of results which are different, even with the same cluster count, whereas the results obtained from the k-means clustering only had different optimal cluster count between clustering indicators. We solved that issue by picking the result with the statistical mode of all cluster count indicated. However, we were not able to use that same procedure for MST with minimal cut, as it provided different results even for the same cluster count. Hence, without an additional procedure that picks out the best answer of MST with minimal cut, k-means solution is more reliable due to uniform answer if given cluster count.

Similar to synthetic data results, we were able to find that most of the results from MST with minimal cut shares the same highly connected APC clusters with k-means clustering's results on the same biological dataset. However, specifically the results from PF00033, MST with minimal cut was only able to find the subset of the highly connected APC cluster as shown from the k-means's results. This might have been caused by MST with minimal cut being aggressive in finding an local optimal solution, neglecting the APC that didn't increase local closeness but would have included if compared to all other clusters. Instead, those APCs were included in k-means clustering because it optimizes the clustering configuration of all of the APCs. This characteristics of k-means is particularly useful if we want to observe more than just one of the cluster configurations, and hence, is preferred over MST with minimal cut.

The results from both the synthetic and biological datasets shows that k-means clustering algorithm is more suitable for our methods and the types of datasets we have applied. In addition, with the runtime advantage that k-means clustering algorithm provides, k-means is used to cluster and find APC clusters that are important in the protein family.

5. CONCLUSION

We conclude that k-means clustering algorithm is more suitable for our problem and datasets than MST with minimal cut. The reasons are due to faster runtime, global optimization, and rendering more consistent final results. Results from synthetic shows that the two clustering algorithms have a positive linear relationship, meaning that their results would not differ much. Results from biological datasets highlighted that MST with minimal cut had more final results and hence are less consistent results to k-means clustering. In addition, global optimization of clusters was preferred in the datasets over local optimization based on graph edges. We analyzed the runtime and found that k-means clustering algorithm was faster than MST with minimal cut. Because k-means clustering algorithm was a more suitable clustering algorithm for our problem and dataset, we verified results from k-means clustering using three-dimensional structure. This algorithm is further used to verify co-occurring APCs in a crystallizing protein related to immunology.

6. REFERENCES

[1] T. K. Attwood, P. Bradley, D. R. Flower, A. Gaulton, N. Maudling, A. Mitchell, G. Moulton, A. Nordle, K. Paine, P. Taylor, et al. Prints and its automatic supplement, preprints. *Nucleic acids research*, 31(1):400–402, 2003.

[2] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucleic acids research*, 28(1):235–242, 2000.

[3] D. L. Davies and D. W. Bouldin. A cluster separation measure. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2):224–227, 1979.

[4] J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. 1973.

[5] R. D. Finn, J. Mistry, J. Tate, P. Coggill, A. Heger, J. E. Pollington, O. L. Gavin, P. Gunasekaran, G. Ceric, K. Forslund, et al. The pfam protein families database. *Nucleic acids research*, 38(suppl 1):D211–D222, 2010.

[6] A. P. Francisco, S. Schbath, A. T. Freitas, and A. L. Oliveira. Using graph modularity analysis to identify transcription factor binding sites. In *Bioinformatics and Biomedicine Workshops (BIBMW), 2010 IEEE International Conference on*, pages 19–26. IEEE, 2010.

[7] A. R. Kinjo and H. Nakamura. Composite structural motifs of binding sites for delineating biological functions of proteins. *PloS one*, 7(2):e31437, 2012.

[8] E.-S. A. Lee and A. K. C. Wong. Revealing binding segments in protein families using aligned pattern clusters. *Proteome Science*, 2013.

[9] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Minig*. Addison-Wesley, 2006.

[10] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 577–584, 2001.

[11] H. Wolda. Similarity indices, sample size and diversity. *Oecologia*, 50(3):296–302, 1981.

[12] A. K. Wong, D. Zhuang, G. C. Li, and E.-S. Lee. Discovery of delta closed patterns and noninduced patterns from sequences. *Knowledge and Data Engineering, IEEE Transactions on*, 24(8):1408–1421, 2012.