Discovery of Spatially Cohesive Itemsets in Three-dimensional Protein Structures*

Cheng Zhou, Pieter Meysman, Boris Cule, Kris Laukens and Bart Goethals

Abstract—In this paper we present a cohesive structural itemset miner aiming to discover interesting patterns in a set of data objects within a multidimensional spatial structure by combining the cohesion and the support of the pattern. We propose two ways to build the itemset miner, *VertexOne* and *VertexAll*, in an attempt to find a balance between accuracy and run-times. The experiments show that *VertexOne* performs better, and finds almost the same itemsets as *VertexAll* in a much shorter time. The usefulness of the method is demonstrated by applying it to find interesting patterns of amino acids in spatial proximity within a set of proteins based on their atomic coordinates in the protein molecular structure. Several patterns found by the cohesive structural itemset miner contain amino acids that frequently co-occur in the spatial structure, even if they are distant in the primary protein sequence and only brought together by protein folding. Further various indications were found that some of the discovered patterns seem to represent common underlying support structures within the proteins.

Index Terms—itemset mining, multidimensional data, cohesion, protein structure

1 Introduction

Pattern discovery in sequences is a popular data mining task. Usually, a pattern is evaluated based on how close to each other its elements occur (cohesion), and how often the pattern itself occurs (support). Recently, attempts have been made to mine interesting patterns in sequences by combining cohesion and support [2]. Here we extend this method into data objects with a multidimensional structure and explore its potential to find interesting amino acid patterns within a set of proteins based on their atomic coordinates and molecular structure information.

Proteins are linear chains composed of twenty different amino acids (often referred to as 'residues'). In living cells these chains fold into specific three-dimensional structures that perform a great variety of biological functions. In the structure of a single protein we distinguish the primary structure, which corresponds to the sequence of the amino acids as they occur along the protein chain; the secondary structure, which is a local shape, such as α-helices or β-sheets, adopted by small segments of consecutive amino acids; the tertiary structure, which is the complete three-dimensional structure of the protein; and the quaternary structure, which corresponds to intermolecular interactions that proteins undergo. There is a vast amount of molecular structure data publicly available in biological databases. The RCSB Protein Data Bank (PDB), which is the single worldwide repository of molecular structures of large biological molecules, currently contains the three-dimensional atomic coordinates of more than 90 000 structures [3]. Although the discovery of conserved structural motifs in proteins is a widely explored field in bioinformatics, the majority of protein pattern mining algorithms focus on the sequence dimension and do not consider other spatial dimensions. The extraction of spatial patterns can potentially reveal significant biological insights into the properties of different protein classes. The discovery of patterns within the tertiary structure of proteins unavoidably requires advanced computational algorithms due to its dimensionality.

There are several tools available for analysing protein structures, either for calculating similarities between whole or parts of the structures, or finding features that can assist in the problem of protein structural annotation and the classification of protein functions [4], [5], [6]. However, the discovery of interesting patterns or arrangements of amino acids within a large structural data set in an unsupervised and rapid manner remains an important research objective. Here we explore the concept of cohesion for high dimensional itemset mining to extract sets of amino acids that frequently spatially co-occur in a given set of three-dimensional protein structures.

The characterisation of amino acids that are in close proximity to each other within a protein structure is somewhat similar to the purpose of protein contact maps. These maps are two-dimensional matrices detailing the pairwise inter-residue contacts of a protein, where a contact between two amino acids is defined if the distance between them is lower than

- *The authors are with the Department of Mathematics and Computer Science, University of Antwerp, Belgium.*
  E-mail: {cheng.zhou, pieter.meysman, boris.cule, kris.laukens, bart.goethals}@uantwerpen.be
- *C. Zhou is also with the National University of Defense Technology, China.*
- *P. Meysman and K. Laukens are also with the Biomedical informatics research center Antwerpen (biomina), Belgium.*

** A preliminary version appeared as "Mining Spatially Cohesive Itemsets in Protein Molecular Structures", in the Proceedings of the 12th International Workshop on Data Mining in Bioinformatics (BioKDD'13) [1].*

a given threshold. The construction of such a contact map is a common step in the *ab initio* prediction of the full molecular structure of a protein from its sequence [7]. While itemset mining techniques have been successfully applied to such protein contact maps, the primary goal of these studies remained the improvement of *ab initio* prediction [8]. Other mining algorithms for finding common amino acid patterns involve the conversion of the protein structure to a graph representation prior to subgraph mining [9], [10], [11], [12].

The goal of this paper is to explore whether cohesive structural itemset mining can reveal potentially interesting biological patterns. The type of interaction explored differs greatly from previous methods based on contact maps or from graph representations. Firstly, the presented algorithm directly mines the three-dimensional co-ordinates of the amino acids and thus suffers no loss of information due to a conversion to a two-dimensional space or to a graph. Secondly, the recent development of the cohesion concept allows the algorithm to mine the data without setting a cut-off on the maximum distance in which relationships between amino acids can occur. This potentially allows the discovery of relationships where the amino acids are not in direct contact, such as, for example, residues forming a metal-binding site. Thirdly, the application of itemset mining on the protein structure itself allows discovery of patterns that concern several amino acids, instead of the pairwise combinations of amino acids such as in contact maps or distance matrices.

The rest of the paper is organised as follows. We formally describe the problem setting for finding spatially cohesive itemsets in Section 2. In Section 3, we present our algorithm for generating interesting itemsets. We end the paper with an experimental evaluation in Section 4 and a summary of our conclusions in Section 5.

2 Problem Setting

We consider a data object with an n-dimensional structure as a list of points where a point v is a pair (a, c) consisting of an item $a \in I$ and an n-dimensional coordinate $c \in \mathbb{R}^n$, where I is the set of all possible items and $n \geq 1$. Clearly, two points can never occur at the same position, i.e. with the same coordinate. On the other hand, an item a_i may occur many times at different positions in a data object d_g. Thus there may be many points containing a_i in d_g and we denote such points as V_{gi}. Here, we denote a data object by $d = \{v_1, \ldots, v_l\}$, where l is the number of points in the data object. A database *DB* is a set of data objects. The set of all data objects in *DB* is denoted by D.

The patterns considered in this paper are itemsets, or sets of items coming from the set I. The support count of an itemset is defined as the number of different data objects in which the itemset occurs, regardless of how many times the itemset occurs in any single data object. In other words, when looking for the support count of a single itemset, we can stop looking at a data object as soon as we have encountered the first occurrence of the itemset in that data object.

To determine the interestingness of an itemset, however, it is not enough to know how many times the items making up the itemset occur. In this paper, we are specifically investigating patterns of items occurring spatially in close proximity. To do this, we will define interesting itemsets in terms of both support and cohesion.

2.1 Support

For a given itemset X, we denote the set of data objects that contain all items of X as $N(X) = \{d \in D | \forall a \in X, \exists (a, c) \in d\}$. The *support* of X in database *DB* can now be defined as

$$S(X) = \frac{|N(X)|}{|D|}. \quad (1)$$

2.2 Cohesion

Given a set of points $V = v_1, \cdots, v_q$, let $MB(V)$ denote the ball with the smallest radius that contains V, namely the *smallest enclosing ball*. It has been shown that $MB(V)$ always exists and is unique [13]. Intuitively, we consider the points V in n-dimensional space cohesive if the radius of $MB(V)$ is small enough.

Given an itemset $X = \{a_1, \ldots, a_m\}$, assume that each item a_i occurs n_i times in a given data object $d_g \in N(X)$. If we wish to find the exact smallest enclosing ball of X in d_g, there are $\prod_{i=1}^{m} n_i$ combinations for each of which we need to find the smallest enclosing ball, and then find the one with the minimal radius. This process is time consuming. As a result, we propose two different ways to approximate this process, *VertexOne* and *VertexAll*.

2.2.1 VertexOne

Intuitively, points that occur near each other are more likely to produce the smallest enclosing ball than those far apart. Therefore, rather than looking at all possible combinations of points, we will limit our search to a selection of points. In our first approach, we approximate the process of finding the smallest enclosing ball of an itemset as follows:

1. select an item a_1 from $X = \{a_1, \ldots, a_m\}$ (all the items are sorted by descending support), and for each point $v_j \in V_{g1},\ j = 1, 2, \ldots, n_1$, we find the nearest point in each set of points of other items in X, namely $V_{g2}, \ldots, V_{gm}$. We thus obtain the set of nearest points

$$NV_j = \{v | v = \arg\min_{w \in V_{gi}} D(w, v_j),\ i = 2, 3, \ldots, m\}, \quad (2)$$

where $D(w, v_j)$ is the Euclidean distance between point w and point v_j.

2. we denote a nearest combination as $B_j = \{v_j\} \cup NV_j$ and $B = \{B_j | j = 1, 2, \ldots, n_1\}$, so we get n_1 nearest combinations.

3. for each set $B'_j \in B$, find $MB(B'_j)$ and get its radius $R'_j(X)$.

4. denote the smallest radius in a given data object $d_g \in N(X)$ as

$$R'_g(X) = \min_{j=\{1,\ldots,n_1\}} R'_j(X). \quad (3)$$

There are only n_1 smallest enclosing balls to find in a data object d_g, much fewer than if we tried to find the exact smallest enclosing ball of X in d_g, resulting in a considerable reduction in time complexity. We sort the items by descending support to get more combinations of points, in order to reduce the resulting approximation error.

In the worst case, the smallest radius we find here could be nearly twice as large as the exact radius of the smallest enclosing ball containing items of an itemset X, as illustrated in Figure 1. In this simple two-dimensional example, assume we are evaluating itemset abc, and we picked item a as the starting point. We look for the nearest b and the nearest c, and find the only b, and c_1, which is closer to a than c_2, resulting in the ball drawn with a dashed line. However, the smallest possible ball containing a, b and c is much smaller, and is depicted using a solid line.

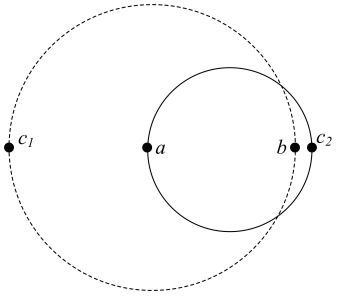

Fig. 1: An Example of *VertexOne*

2.2.2 VertexAll

As can be seen above, *VertexOne* can, in theory, result in large approximation errors. We therefore develop another way to increase the probability of getting the exact smallest enclosing ball as follows.

1. for each item a_i in $X = \{a_1, \ldots, a_m\}$ (all the items are sorted by ascending support), we perform steps 1 and 2 of *VertexOne*, and obtain N nearest combinations, where $N = \sum_{i=1,\ldots,m} n_i$.

2. for each combination $B''_j, j = 1, 2, \ldots, N$, find its smallest enclosing ball and get its radius $R''_j(X)$.

3. then denote the smallest radius in a given data object $d_g \in N(X)$ as

$$R''_g(X) = \min_{j=\{1,\ldots,N\}} R''_j(X). \quad (4)$$

There are now N smallest enclosing balls to find in a data object d_g, which will further limit the approximation error, but the time complexity is now higher than that of *VertexOne*. Note that the output of *VertexAll* will be independent of the order in which the items are sorted. However, we sort the items by ascending support in order to reduce the search space and runtime.

The worst case error made by *VertexAll* is dependent on the size of the itemset. Figure 2 shows the worst case for an itemset of size three, where the smallest possible ball containing a, b and c is depicted using a solid line. However, searching from any given point, we will find a ball with a radius nearly $\sqrt{3}$ times as large as the exact radius of the smallest possible ball. For example, starting off from point a_1, we will find the nearest combination a_1, b_1, c_2, resulting in the ball drawn with a dashed line at the top left of Figure 2. Similar results come out for other points. We can construct similar worst case data objects for larger itemsets, which lead us to two observations:

1. Given an itemset $X = \{a_1, \ldots, a_m\}$, the worst case data object will result in discovered smallest balls that are centrosymmetric and axisymmetric, and will contain points that form a regular polygon with m sides, that would form an exact smallest enclosing ball (an example is shown in Fig. 2).

2. The smallest radius we find by *VertexAll* could be nearly twice as large as the exact radius of the smallest enclosing ball containing all items of X when m approaches infinity.

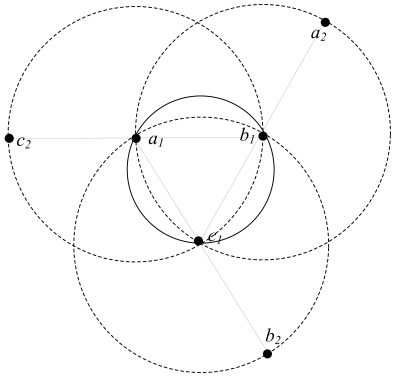

Fig. 2: An Example of *VertexAll*

2.2.3 Cohesive Radius

To evaluate the cohesion of an itemset X in the whole dataset, we need to compute the smallest radius $R_g(X)$ in each data object d_g that contains X. We define the *cohesive radius* of X in D as

$$R(X) = \frac{\sum_{d_g \in N(X)} R_g(X)}{|N(X)|}, \quad (5)$$

where $R_g(X)$ is either $R'_g(X)$ or $R''_g(X)$, depending on whether we use *VertexOne* or *VertexAll*.

2.3 Interesting Itemset

Given a minimum support threshold *min_sup* and a maximum cohesive radius threshold *max_rad*, X is an *interesting itemset* if $S(X) \geq min\_sup$ (X is frequent) and $R(X) \leq max\_rad$ (X is cohesive). Note that the smaller the radius $R(X)$ the higher the cohesion of X.

3 Generating the Complete Set of Interesting Itemsets

In this section we present an algorithm for mining interesting itemsets in a database consisting of data objects, each of which containing a number of multi-dimensional points.

3.1 Pruning the Search Space

Note that the cohesive radius of an itemset is not a monotonic measure. In other words, in rare cases, it is possible for the cohesive radius of a smaller itemset to be greater than the cohesive radius of one of its supersets. Consider the following simple example. Assume that the dataset consists of just three data objects, d_1 and d_2, containing items a, b and c, and d_3, containing only items a and b. It is perfectly possible that the radii of the smallest balls containing itemset abc in both d_1 and d_2 are smaller than the radius of the smallest ball containing itemset ab in d_3. In this case, $R(abc)$ (the cohesive radius of itemset abc, as defined in Equation 5) will be smaller than $R(ab)$, even though ab is a subset of abc.

Although the cohesive radius of an itemset is not monotonic, we can still use its properties for pruning certain candidates from the search space. Our pruning method for *VertexOne* is based on two observations:

1. If itemset X is a subset of itemset Y, and they both occur in a data object d_i, then $R'_i(X) \leq R'_i(Y)$.

2. Given a minimum support threshold *min_sup*, an itemset must occur in at least $\lceil min\_sup \times |D| \rceil$ data objects to be frequent. Assume that itemset X occurs in k data objects, with $k \geq \lceil min\_sup \times |D| \rceil$, and sort these data objects such that $R'_1(X) \leq \ldots \leq R'_k(X)$. For any frequent itemset Y that is a superset of X, it holds that

$$R'(Y) \geq \frac{\sum_{i=1,\ldots,\lceil min\_sup \times |D| \rceil} R'_i(X)}{\lceil min\_sup \times |D| \rceil} = LBR'(X).$$

In other words, $LBR'(X)$ as defined above, can serve as a lower bound for the cohesive radius of all frequent supersets of X. As a result, if X is frequent, but its cohesive radius is large enough, we can be sure that none of its supersets can be both frequent and cohesive.

Similarly, we also find a lower bound for the cohesive radius of a superset using the *VertexAll* method:

1. Given a data object d_g, and itemsets X and Y, such that $X \subset Y$, if both X and Y occur in d_g, it does not hold that $R''_i(X) \leq R''_i(Y)$, as was the case for *VertexOne*. The cohesive radius can now actually get smaller if an item is added to the itemset that allows us to find a smaller smallest ball. Consider the example given in Figure 2. If we add a point d inside the solid circle, $R''_i(abcd)$ found by *VertexAll* will be the radius of the solid circle which is much smaller than $R''_i(abc)$ (one of the dashed circles). However, thanks to the worst case analysis given in Section 2.2.2, we still know that the cohesive radius of a superset will satisfy the following inequality: $R''_i(X) \leq 2R''_i(Y)$.

2. Following the same line of reasoning we used for *VertexOne*, we can conclude that for any frequent itemset Y that is a superset of X, it holds that

$$R''(Y) \geq \frac{\sum_{i=1,\ldots,\lceil min\_sup \times |D| \rceil} R''_i(X)}{2\lceil min\_sup \times |D| \rceil} = LBR''(X).$$

As can be seen, the lower bound for *VertexAll* is not as strict as that of *VertexOne*. The denominator is now twice as large, but the numerator now uses a more precise $R''(X)$ radius rather than the $R'(X)$ version used in *VertexOne*. Experiments show that $R'(X)$ is virtually never twice as large as $R''(X)$, and, as a result, *VertexAll* will prune a lot less.

3.2 Main Algorithm

In this section we give a description of our main algorithm, which is the same regardless of whether we use *VertexOne* or *VertexAll*. In further text we therefore use $R(X)$ to denote $R'(X)$ and $R''(X)$, and $LBR(X)$ to denote $LBR'(X)$ and $LBR''(X)$, respectively.

Our algorithm generates all interesting itemsets in two steps. In the first step, we use an Apriori-like algorithm to find the frequent itemsets. In the second step, we determine which of the frequent itemsets are actually spatially cohesive and utilise the observations above to prune the itemsets that cannot be both frequent and cohesive.

Let *n-itemset* denote an itemset of size n. Let F_n denote the set of frequent *n-itemset*s. Let C_n be the set of candidate *n-itemset*s and T_n be the set of interesting *n-itemset*s. The algorithm for generating the complete set of interesting itemsets in a given set of data objects D is shown in Algorithm 1. Two optional parameters, min_size and max_size, can be used to limit the output only to interesting itemsets with a size bigger than or equal to min_size and smaller than or equal to max_size.

Lines 1-5 count the supports of all the items to determine the interesting 1-itemsets. Since the cohesive radius of a singleton is always equal to 0, if *min_size* ≤ 1, all frequent singletons are stored as interesting. Lines 6-23 discover all interesting itemsets of different sizes n (*max_size* $\geq n \geq 2$). First, the already discovered candidates of size $n-1$ (C_{n-1}) are used to generate the candidate itemsets C_n using the candidateGen function (line 10). The candidateGen function is similar to the function Apriori-gen in the

Algorithm 1 GENERATINGITEMSETS. An algorithm for generating all interesting itemsets in a dataset D.

Require: dataset D, minimum support threshold *min_sup*, maximum cohesive radius threshold *max_rad*, minimum size constraint min_size and maximum size constraint max_size.
Ensure: all interesting itemsets T.

1: $C_1 = \{a | a \in I\}$, I is the set of all items occurring in D
2: $F_1 = \{f | f \in C_1, S(f) \geq min\_sup\}$
3: **if** $1 \geq min\_size$ **then**
4: $T_1 = F_1$
5: **end if**
6: $C_1 = F_1$
7: $n = 2$
8: **while** $C_{n-1} \neq \emptyset$ and $n \leq max\_size$ **do**
9: $T_n = \emptyset$
10: C_n = candidateGen(C_{n-1})
11: $F_n = \{f | f \in C_n, S(f) \geq min\_sup\}$
12: $C_n = \emptyset$
13: **for all** frequent itemset f in F_n **do**
14: **if** $LBR(f) \leq max\_rad$ **then**
15: $C_n = C_n \cup \{f\}$
16: **if** $n \geq min\_size$ and $R(f) \leq max\_rad$ **then**
17: $T_n = T_n \cup \{f\}$
18: **end if**
19: **end if**
20: **end for**
21: $n++$
22: **end while**
23: $T = \bigcup_{i=1}^{n-1} T_i$
24: **return** T

Apriori algorithm [14]. Note that the items are ranked by descending support for *VertexOne* while ranked by ascending support for *VertexAll*. In line 11, we store the frequent itemsets from C_n into F_n. In lines 13-15, we prune the candidates that cannot be both frequent and cohesive, while in lines 16-17, we store the interesting itemsets (as defined in Section 2) from F_n into T_n. The final set of all interesting itemsets in D is stored in T and produced as output.

The two most time consuming steps are the candidate generation and the evaluation of the cohesive radius. For these two steps we use the Apriori algorithm [14] to generate candidates, and an existing implementation[1] of the algorithm for computing the smallest enclosing ball [13], respectively. The time complexity of these algorithms has been extensively analysed in the papers that originally proposed them. Since the smallest enclosing ball must be computed only for itemsets that have been found to be frequent, the runtime will be proportional to the number of generated candidate itemsets.

1. http://www.inf.ethz.ch/personal/gaertner/miniball.html

Since we are using an average over a large number of data objects, the effect of the approximation error will be amortised. For an itemset of size 2, we will always find the exact smallest ball, and for itemsets of size 3 or bigger, the chance of the worst-case error occurring (as described above) decreases as the size of the itemset grows. On two of the small datasets (*Winged* and *Lambda*) we used in our experiments (see Section 4 for more details), it was possible to compute the exact smallest enclosing balls in acceptable runtime. We set the minimum support threshold to 0.8 and the maximum cohesive radius threshold to 4 and 3 angstrom for *Winged* and *Lambda*, respectively. Table 1 shows the average error made by our algorithms. $|Out|$ denotes the number of interesting itemsets we get. The reported average error was obtained by dividing the sum of all relative errors with the total number of the computed smallest balls. As can be seen in the run-times reported in Table 1, the complexity of the exact algorithm is prohibitive on large datasets, while the average error of the approximate algorithms is kept within reasonable limits. In this small example, we can see that *VertexOne* misses out on less than 4% of the patterns we would discover using the exact method, which would take more than $5\,000$ times longer to complete the search. *VertexAll* found the patterns missed by *VertexOne*, but it also took much longer than *VertexOne*.

TABLE 1: The Comparison of Methods

Dataset	Method	$\|Out\|$	Run-time	Average error
Winged	*VertexOne*	159	1.218s	0.01825
	VertexAll	164	400.197s	0.00079
	Exact	164	8217.840s	0
Lambda	*VertexOne*	142	1.437s	0.01546
	VertexAll	145	500.692s	0.00089
	Exact	145	6824.046s	0

4 EXPERIMENTS

The cohesive structural itemset miner was applied to extract patterns from real biological datasets, namely protein molecular structures. The structural information on these proteins was extracted from the PDB public archive [3]. PDB contains the atomic coordinates and molecular structure information for various proteins and other biological macromolecules. The relative locations of the atoms to each other within these molecules were determined by a variety of methods, such as X-ray crystallography, NMR spectroscopy and cryo-electron microscopy. These three-dimensional coordinates of the amino acids of a set of related proteins will make up the backbone of our analysis.

For the purposes of applying the methodology on a wide range of data, four sets of proteins were collected. Two smaller datasets consisted of the proteins annotated by SCOP as containing the

'winged helix DNA-binding domain' (*Winged*) and the 'lambda repressor-like DNA-binding domain', respectively (*Lambda*) [15]. As an additional constraint on these smaller datasets, only structures reporting both the protein and the DNA structure were utilised. Thus only proteins known to be in their active and bound state are considered during the mining process as the free-floating potential inactive state may display considerable differences in its conformation. This approach guarantees the uniformity of the structures to evaluate in these datasets. Two larger sets were based on the molecular function of the protein. To this end, using their gene ontology molecular function annotations, one set of proteins with 'kinase activity' (*Kinase*) and another set with 'peptidase activity' (*Peptidase*) were collected [16]. These datasets therefore represent a wide diversity of proteins that each share a common molecular function. In cases where multiple macromolecules were present in the same PDB entry, only one protein was presented to the algorithm, i.e., the one with a description matching certain keywords (e.g., trypsin or protease for the peptidase set) or the protein with a description similar to the title of the stored structure. In cases of ambiguity (e.g., for k-mer proteins), the first reported protein matching the above criteria was selected.

From the reported protein molecular structure only the position of the α-carbon atom of the amino acid was considered. This atom is present in every amino acid and is the carrier of the side chain unique to each type of amino acid. Each Cα was then labelled with the three-letter name of the corresponding amino acid. This label was further extended with the secondary structure information, which is also included in most PDB structures. The secondary structure concerns the local shape of the amino acids, and a collection of residues within a single protein can form an α-helix (denoted in the itemsets as X_H), a β-sheet (X_B) or a loop of unstructured amino acids (X_U). The addition of the secondary structure to the label is not necessary for the operation of the algorithm, but it is advantageous for the presented experiments. Firstly, the vast majority of proteins contain at least one copy of each amino acid in their sequence and thus without this addition every possible itemset will likely be frequent. Secondly, including this information in the label greatly aids in the interpretability of the found cohesive patterns as many known protein motifs or common structures are expressed in terms of the secondary structure. The input data thus consisted of the (x, y, z) coordinates of the Cα atom labelled by the corresponding amino acid and the secondary structure. In this manner, a protein is converted to a list of points where a point v is a pair (a, c) consisting of the label $a \in I$ and a three-dimensional coordinate $c \in \mathbb{R}^3$, where I is the set of all possible labels (in our case, amino acids). The algorithms presented in Section 3 were then used to generate the interesting itemsets found across these proteins, with each itemset representing a pattern of spatially co-occurring amino acids.

Table 2 shows the run-times of our two algorithms on the four datasets with *min_sup* fixed at 0.8, *max_rad* fixed at 4 angstrom, *min_size* set to 1 and *max_size* unlimited. The third column $|D|$ contains the number of proteins in the datasets, while $|C|$ denotes the number of generated candidates and $|Out|$ denotes the number of generated interesting itemsets. All experiments are performed on a laptop computer with Intel i7 (2 CPUs 2.7GHz), 4GB memory and Windows 7 Professional. From the table, we can see that the run-time largely depends on the number of proteins in the dataset and the number of candidate itemsets. This matches the conclusions of the time complexity analysis performed in Section 3.

TABLE 2: Run-times of The Algorithms

Method	Dataset	$\|D\|$	$\|C\|$	Runtime	$\|Out\|$
VertexOne	*Lambda*	47	579	3.655s	430
	Winged	62	235	1.218s	159
	Kinase	2749	770	378.156s	440
	Peptidase	2558	416	184.269s	226
VertexAll	*Lambda*	47	51011	722.456s	455
	Winged	62	11790	87.658s	164
	Kinase	2749	241399	314285.951s	450
	Peptidase	2558	55470	58791.770s	237

A detailed examination of the additional patterns found by the *VertexAll* algorithm reveals that the majority are simply novel combinations of the items in the patterns that were also found by the *VertexOne* algorithm. Thus they mostly do not describe novel information but another viewpoint on the same. Furthermore, all of the patterns unique to the result of the *VertexAll* algorithm could also be found by the *VertexOne* algorithm, using a higher cohesive radius threshold. Fig. 3 shows the patterns found only by the *VertexAll* variant for the four datasets using the setup described in the respective subsections below. We then ran *VertexOne* again with a higher cohesive radius threshold and found the missing patterns. For each unique pattern the ranking based on its cohesion score with respect to all other patterns is provided for both the *VertexAll* and the *VertexOne* algorithm. With only a few exceptions, the ranking of the patterns resulting from either variant was largely the same, i.e. they occur on or around the diagonal in the plot. None of the patterns found only by *VertexAll* were ranked in the top 50 most cohesive itemsets. Therefore, the most frequent and cohesive patterns in all of the datasets were identical in the results of the *VertexAll* and the *VertexOne* algorithm, with only a small difference in the actual cohesive radius reported. As we choose not to provide an exhaustive examination of all the found patterns but only discuss the most cohesive patterns in each dataset, which are thus independent from the used methodology, we will base our findings on the

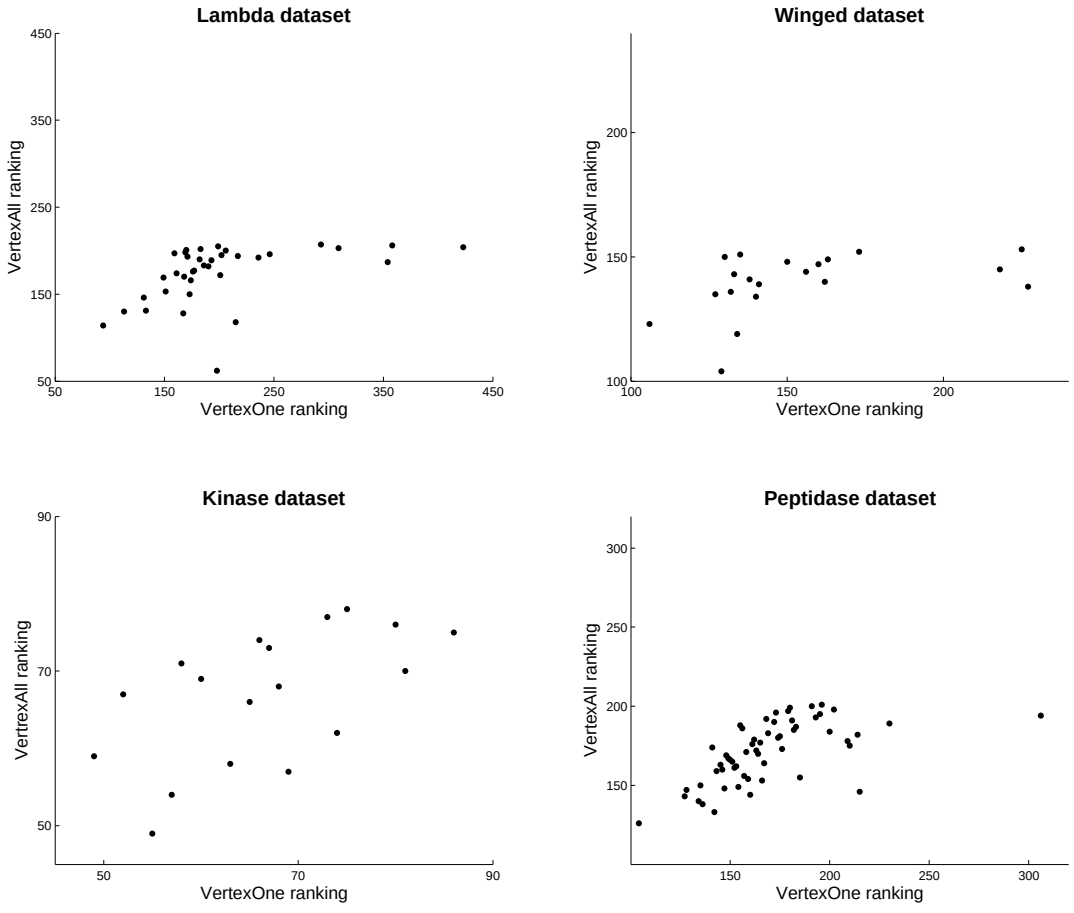

Fig. 3: Ranking Comparison between *VertexOne* and *VertexAll* for the patterns originally missed by *VertexOne*. In order to make the comparison, we ran *VertexOne* again with a higher cohesive radius threshold.

results of the *VertexOne* variant.

The maximal cohesion radius parameter has a large impact on the output of the algorithm. Table 3 shows a summary of the results of *VertexOne* on the four datasets with different *max_rad* thresholds. Here, we set *min_sup* to 0.8, *min_size* to 3 and *max_size* to unlimited. $|Out-n|$ denotes the number of interesting itemsets of size n we get. As can be seen in Table 3, the algorithm finds more interesting itemsets but has longer run-times with larger cohesive radius thresholds. There is also a large dependency between the *max_rad* threshold and the size of the itemsets that can be found. At the lower cohesion radius values, most cohesive patterns consist of three amino acids. As the radius increases, so does the size of the found patterns. In the *Lambda* dataset, the algorithm finds patterns with up to nine amino acids for a maximal cohesion radius of 8 angstrom. This dependency is likely due to the trade-off between adding additional amino acids to the itemset and a resulting decrease in cohesion and frequency of the pattern. Indeed, due to the steric constraints of amino acid placement, one can expect that adding a single amino acid would have a great effect on the cohesive radius of any pattern. Only the results for the smaller cohesion radius values will be discussed in detail in the next sections as these concern the most cohesive patterns found for each data set.

4.1 Lambda Repressor-like Proteins

The first small dataset the algorithm was applied to consists of 47 proteins annotated with a lambda repressor-like DNA-binding domain. This set therefore consists mostly of transcription factors, which are DNA-binding proteins that regulate the expression of downstream genes. The archetypical protein for this type of domain is the bacteriophage lambda C1 repressor, which is a viral regulator [17]. Several proteins containing a lambda repressor-like domain are of great biological importance and the mechanism by which such proteins interact with the DNA molecule are well understood. For example, the lactose repres-

TABLE 3: Impact of the Maximal Cohesion Radius on the Experimental Results

Dataset	*max_rad*	Runtime	$\|Out-3\|$	$\|Out-4\|$	$\|Out-5\|$	$\|Out-6\|$	$\|Out-7\|$	$\|Out-8\|$	$\|Out-9\|$
Lambda	4	3.484s	165	17	0	0	0	0	0
	5	7.834s	464	226	9	0	0	0	0
	6	19.492s	1049	871	331	37	0	0	0
	7	53.812s	1694	2535	1644	641	128	7	0
	8	148.365s	2117	5400	5947	3206	1038	197	15
Winged	4	1.192s	21	0	0	0	0	0	0
	5	2.404s	131	5	0	0	0	0	0
	6	4.590s	342	151	2	0	0	0	0
	7	9.771s	722	542	188	4	0	0	0
	8	21.851s	1268	1444	885	288	21	0	0
Kinase	4	331.833s	68	0	0	0	0	0	0
	5	903.904s	477	24	0	0	0	0	0
	6	3322.579s	1953	484	8	0	0	0	0
	7	12478.511s	4509	3962	726	19	0	0	0
	8	63734.033s	5880	18161	9747	1775	87	0	0
Peptidase	4	137.232s	27	0	0	0	0	0	0
	5	314.354s	333	4	0	0	0	0	0
	6	969.114s	822	426	4	0	0	0	0
	7	2858.497s	1361	1869	638	8	0	0	0
	8	8179.023s	1692	4353	4241	1546	61	0	0

sor (LacI) is commonly used as a model for transcriptional regulation and the interaction between LacI and its binding sites has been the subject of intensive study over the past several decades [18]. The typical lambda repressor-like domain consists of four α-helices in a closed leaf motif. This protein dataset is therefore an ideal case study to evaluate if the patterns uncovered through the presented methodology can be related to known biological significance.

The cohesive structural itemset miner was applied to these protein structures to find amino acids that were consistently grouped in close proximity across a large fraction of the proteins. The reported patterns were filtered based on their uniqueness to a specific dataset at a support cut-off of 80%. The cohesive radius threshold was set to 4 angstrom, *min_size* to 3, and *max_size* unlimited. The most cohesive patterns specific for the lambda repressor-like proteins can be found in Table 4 with their respective cohesive radius in angstrom and support. A total of 182 patterns were found within the set thresholds, of which 165 were itemsets containing three amino acids, while the other 17 contained four amino acids.

It is apparent from the labels of the extracted itemsets that most describe amino acids in α-helices. This can be expected as the annotated domain used to create this dataset, namely the lambda repressor like-DNA binding domain, consisted mostly of α-helices. Additionally, amino acids within a single α-helix can be expected to be frequently co-occurring due to the intrinsic shape of protein helices. However, a comparison between the found itemsets and an alignment of the protein amino acid sequences reveals that not all patterns are limited to the conserved region between these proteins. In the next step, the locations of the itemsets within the protein structure are visualised to give an overview of their distribution throughout the structure.

TABLE 4: The 30 Most Cohesive Patterns For The *Lambda* Dataset

Itemset	Cohesive radius	Support
GLU_H ARG_H ILE_H	2.78	0.80
ALA_H LEU_H MET_H	2.87	0.93
ALA_H GLU_H VAL_H	2.87	0.93
ALA_H ARG_H PHE_H	2.89	0.82
ALA_H ARG_H LYS_H	2.89	0.97
ALA_H GLU_H LYS_H	2.89	0.93
ALA_H GLU_H ASP_H	2.91	0.93
GLU_H VAL_H LYS_H	2.92	0.93
ALA_H VAL_H LYS_H	2.92	0.93
GLU_H ARG_H LYS_H	2.93	0.93
ALA_H GLU_H ARG_H	2.95	0.93
ALA_H GLU_H LEU_H	2.96	0.93
ALA_H LEU_H GLY_H	2.99	0.93
GLU_H ARG_H THR_H	3.00	0.91
ALA_H VAL_H ARG_H	3.01	0.93
ALA_H LEU_H VAL_H	3.02	0.93
VAL_H ARG_H ASN_H	3.04	0.91
ALA_H GLU_H ILE_H	3.04	0.80
ALA_H VAL_H ILE_H	3.11	0.80
VAL_H ARG_H SER_H	3.13	0.93
ALA_H LEU_H PHE_H	3.14	0.82
ALA_H LEU_H ARG_H	3.14	0.97
LEU_H ARG_H ILE_H	3.16	0.85
ALA_H LYS_H ILE_H	3.17	0.85
ALA_H VAL_H ASP_H	3.17	0.93
ALA_H LEU_H TYR_H	3.19	0.93
GLU_H VAL_H ILE_H	3.19	0.80
GLU_H LEU_H MET_H	3.21	0.89
ALA_H GLU_H THR_H	3.24	0.91
ALA_H VAL_H SER_H	3.24	0.93

Fig. 4 shows the protein structure of the *Escherichia coli* PurR repressor (from PDB 1PNR) plotted using the open source version of Pymol. Note that the reported structure in the PDB file only contained one side of the symmetrical protein-DNA complex and thus only features one protein within the protein complex and one DNA strand of the DNA-helix. The atoms of the protein are presented in the stick representation while those of the DNA molecule are reduced to a cartoon representation. The amino acids matching the 171 patterns extracted for the

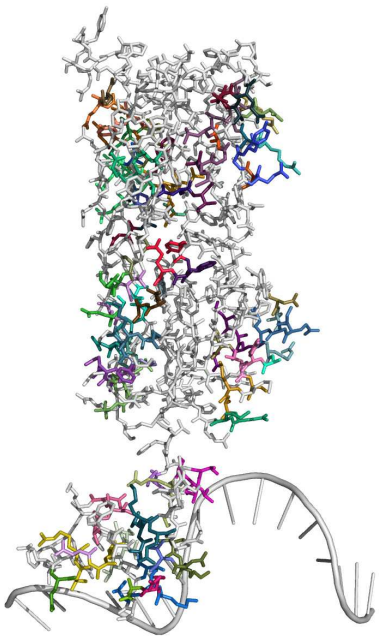

Fig. 4: The Molecular Structure of the PurR protein

lambda repressor-like domain proteins are provided in a colour corresponding to the amino acid content of the pattern, while amino acids not part of any pattern are given in grey. This protein is a bacterial regulator of purine metabolism and is part of the LacI-GalR protein family. This transcription factor is annotated as containing a similar DNA-binding domain as the Lambda C1 repressor on the N-terminal domain, except that it is missing the first α-helix. It also displays a C-terminal domain with a ligand-binding and dimerisation motif similar to the ligand binding sites of periplasmic sugar-binding proteins. The two domains are connected with a hinge sequence that also contains several functional residues for DNA-binding. For example, the leucine present at position 54 in the hinge helix is known to intercalate into the DNA molecule during complex formation causing the induction of a DNA bend [19]. As can be seen in Figure 4, several patterns match amino acids that form the DNA-binding domain. Additionally there are other patterns that are present in the C-terminal domain of the protein or as part of the hinge helix. Inside the hinge helix, most of the amino acids matched up to one or more of the discovered patterns. Several of these patterns include the intercalating leucine residue, such as the pattern ARG_H, ALA_H, LEU_H and VAL_H (i.e. the combination of arginine, alanine, leucine and valine in a helix conformation). As not all lambda repressor-like proteins contain the hinge helix, it is interesting that so many patterns are still found within this segment. Within the DNA-binding domain, there is a notable lack of the central threonine (THR_{16}) residue in any pattern, most likely because this amino acid is missing in several members of the LacI-GalR family. The presence or the absence of threonine at this position in the protein has been proposed to confer differential specificity between LacI-GalR proteins to their DNA targets [20]. Similar findings could be observed for the other proteins within this dataset. Most patterns do not match the amino acids specific for a single protein, which, for example, confer the DNA-binding specificity, but instead match 'supporting' amino acids which seem to be necessary for the overall protein structure and the presentation of the specific residues to the ligands that can be bound by the protein.

4.2 Winged Helix Proteins

The second small dataset contains 62 proteins annotated with a winged-helix DNA-binding domain. The winged-helix domains typically consist of three α-helices, three β-strands forming a twisted antiparallel β-sheet and two large loops or 'wings' [21]. While most proteins present in this set are transcription factors, this set also includes DNA replication initiation proteins (e.g., the F plasmid RepE: PDB 2Z9O), helicases (e.g., *Archaeoglobus fulgidus* Hel308: PDB 2P6R) and endonucleases (e.g., *Planomicrobium okeanokoites* FokI: PDB 1FOK). Thus while these proteins share significant structural similarity, their molecular function is very divergent. In this experiment, the support threshold was set to 80%, *max_rad* to 5 angstrom, *min_size* to 3, and *max_size* unlimited. The application of the presented algorithm to this dataset then resulted in 136 patterns, of which all but five consisted of three amino acids and the remainder of four amino acids. The most cohesive patterns for this dataset can be found in Table 5 with their cohesive radius in angstrom and support. Similar to what was reported for the lambda repressor-like proteins, many of the patterns include amino acids contained within α-helices. Comparison with sequence alignment of the proteins reveals that while several patterns are derived from the α-helices present in the winged-helix domain, the majority of the patterns occur in other segments of the protein.

Fig. 5 shows the molecular structure of the *E. coli* CRP protein (from PDB 1O3T), a transcription factor with a winged helix domain present in the *Winged* dataset. The CRP transcription factor usually binds DNA as a protein complex with two copies of the CRP protein and is known to regulate more than 180 genes, mostly those associated with the carbon metabolism, in *E. coli*. The CRP protein consists of a C-terminal DNA-binding domain containing the winged helix motif and an N-terminal dimerisation domain consisting of β-sheets and a long α-helix. This α-helix is critical for the conformational changes resulting in the activation of CRP induced upon the binding of its ligand, cAMP [22]. In Fig. 5 the CRP dimer bound to its operator site plotted using the open source

TABLE 5: The 30 Most Cohesive Patterns For The *Winged* Dataset

Itemset	Cohesive radius	Support
LEU_H ILE_H ARG_H	3.27	0.88
LEU_H ARG_H SER_H	3.52	0.98
LEU_H ARG_H ALA_H	3.57	0.98
LEU_H SER_H ALA_H	3.63	0.96
LEU_H ILE_H ALA_H	3.72	0.87
LEU_H ARG_H VAL_H	3.75	0.93
LEU_H GLU_H ILE_H	3.78	0.88
ILE_H ARG_H ALA_H	3.83	0.87
LEU_H ARG_H LYS_H	3.84	1.00
LEU_H SER_H VAL_H	3.86	0.93
LEU_H ALA_H VAL_H	3.88	0.91
LEU_H ILE_H VAL_H	3.89	0.82
LEU_H ARG_H TYR_H	3.89	0.87
LEU_H ARG_H ASN_H	3.89	0.9
LEU_H GLU_H GLN_H	3.9	0.96
LEU_H LYS_H PHE_H	3.92	0.85
LEU_H LYS_H VAL_H	3.93	0.93
ARG_H SER_H ALA_H	3.97	0.96
LEU_H LYS_H ALA_H	3.97	0.98
LEU_H ALA_H ASN_H	3.98	0.88
LEU_H LYS_H GLY_H	3.99	0.85
LEU_H ASN_H VAL_H	4.00	0.83
LEU_H GLU_H PHE_H	4.00	0.85
LEU_H ILE_H THR_H	4.02	0.85
LEU_H ARG_H THR_H	4.03	0.91
GLU_H LYS_H VAL_H	4.05	0.93
LEU_H GLU_H VAL_H	4.06	0.93
LEU_H ALA_H PHE_H	4.07	0.85
LEU_H ALA_H THR_H	4.10	0.90
LEU_H GLU_H SER_H	4.13	0.98

version of Pymol. Only one of the two copies forming the protein complex was presented to the cohesive structural itemset miner, namely the one to the left in this figure. The atoms of the protein are presented in the stick representation while those of the DNA molecule and the second CRP copy are reduced to a cartoon representation. The amino acids matching the patterns extracted for the winged helix domain proteins are provided in a colour corresponding to the amino acid content of the itemset, while amino acids that do not match any pattern are given in light grey. The patterns extracted for the entire winged helix protein set concern the amino acids that make up the DNA-binding domain and those contained within the long α-helix directed towards the dimerisation interaction region.

The results for the winged helix proteins with different molecular functions are very similar to those reported above for the CRP protein. The RepE protein involved in the replication initiation of the F plasmid, is known to contain two winged helix domains: one at the N-terminal side of the protein and the other at the C-terminal side. These two domains are separated by a linker region, which accepts a conformational change necessary for dimerisation of RepE [23]. Amino acids present in the winged helix domain and the linker domains match various patterns found in the entire dataset. Several of these patterns, such as ARG_H LEU_H LYS_H (i.e., Arginine, Leucine and Lysine in α-helix conformation), match the LEU_{39} residue

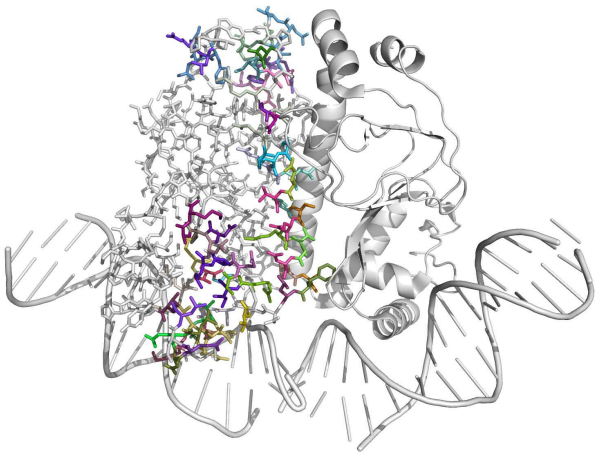

Fig. 5: The Molecular Structure of the CRP Protein

of the RepE which is not part of the dimerisation interface but has been postulated to aid in the correct placement of an α-helix necessary to stabilise the protein dimer [23]. As can be seen in Table 5, the majority of the patterns found for the winged helix proteins contain a leucine amino acid. Given that several leucine residues in RepE act as 'scaffold' amino acids to stabilise the dimer conformation, it seems likely that at least some of the leucine residues within these itemsets perform a similar function in a number of the winged helix domain proteins. Indeed, this corresponds to the results for the CRP protein where the occurrences of the pattern seemed to concern the amino acids responsible for the stabilisation of the dimer structure.

4.3 Kinase Proteins

The first of the larger datasets consists of 2749 proteins displaying kinase activity. These are proteins that catalyse a chemical reaction that transfers a phosphate group to a substrate, a process termed phosphorylation. This substrate is most commonly another protein and phosphorylation may cause conformation change in the substrate protein, for example, causing it to switch from an inactive to an active state. Based on their protein structures and substrates specificity, kinases are divided into the 'protein kinase-like superfamily' and then a set of 'atypical kinases' whose structures greatly differ and can be further subdivided according to common domains [24]. The typical protein kinases share a common catalytic segment consisting of an N-terminal subdomain of mostly β-sheets and a C-terminal subdomain with mostly α-helices. Using a support threshold of 80%, *max_rad* equal to 4 angstrom, *min_size* equal to 3, and *max_size* unlimited, the cohesion-based structural miner resulted in a set of 68 patterns consisting of three amino acids in close proximity. The most cohesive patterns for this dataset can be found in Table 6 with their cohesive radius in

angstrom and support. The majority of the patterns consist of residues within α-helices. Furthermore, in several proteins, these patterns could be directly related to the catalytic regions of the kinase.

TABLE 6: The 30 Most Cohesive Patterns For The *Kinase* Dataset

Itemset	Cohesive radius	Support
LEU_H ALA_H GLU_H	3.23	0.97
ALA_H GLU_H ILE_H	3.28	0.97
LEU_H GLU_H ILE_H	3.28	0.97
LEU_H ALA_H LYS_H	3.36	0.96
LEU_H GLU_H ARG_H	3.39	0.97
LEU_H ALA_H GLY_H	3.40	0.93
LEU_H ALA_H ILE_H	3.41	0.96
LEU_H ALA_H VAL_H	3.45	0.97
LEU_H ALA_H GLN_H	3.46	0.95
LEU_H GLU_H LYS_H	3.46	0.96
LEU_H GLU_H SER_H	3.49	0.96
LEU_H GLU_H VAL_H	3.50	0.97
ALA_H GLU_H ARG_H	3.5	0.97
LEU_H LYS_H ILE_H	3.51	0.96
LEU_H ILE_H GLY_H	3.53	0.93
LEU_H ALA_H PHE_H	3.54	0.94
LEU_H ILE_H ARG_H	3.55	0.96
LEU_H ALA_H ASP_H	3.56	0.95
LEU_H ALA_H ARG_H	3.59	0.97
ALA_H GLU_H VAL_H	3.62	0.98
LEU_H ILE_H VAL_H	3.63	0.97
LEU_H GLU_H ASP_H	3.63	0.96
GLU_H ILE_H ARG_H	3.64	0.97
LEU_H ILE_H GLN_H	3.64	0.94
LEU_H ARG_H GLY_H	3.68	0.93
ALA_H ILE_H ARG_H	3.71	0.97
ALA_H VAL_H ASP_H	3.73	0.96
LEU_H LYS_H ASP_H	3.73	0.95
ALA_H ILE_H VAL_H	3.75	0.97
GLU_H ILE_H VAL_H	3.78	0.97

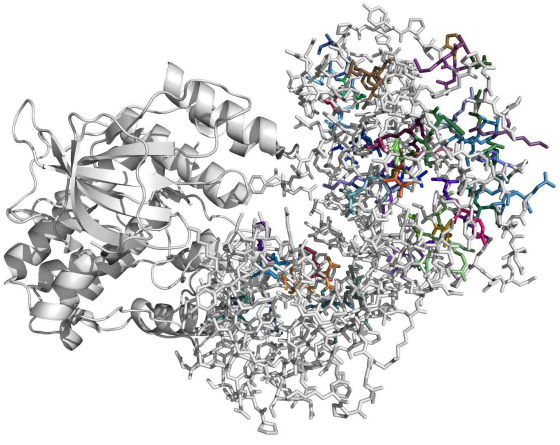

Fig. 6: The Molecular Structure of the Fus3 Protein

An example of a typical protein kinase within our dataset is the *Saccharomyces cerevisiae* MAP kinase, Fus3, which forms an essential part of the mating signalling pathway in yeast. The protein structure contains a C-terminal and an N-terminal region connected by a short hinge section. The catalytic loop containing the functional amino acids for the phosphorylation is contained within the N-terminal region [25]. The molecular structure of Fus3 can be seen in Fig. 6. Only one Fus3 copy from PDB 2F49 was mined for patterns, which is shown in the figure by the stick representation, while the other, shown in cartoon representation, was excluded. The residues that match one of the found patterns for the *Kinase* dataset are annotated in colour. Several patterns were found to describe residues within the catalytic loop of Fus3. These include a pattern describing the amino acids SER_{141} and LYS_{139} within the catalytic loop, and LEU_{100}, which is part of a neighbouring α-helix. The SER_{141} and LEU_{100} residues occur together in these patterns as the spatial distance between their Cα only spanned 5.8 angstroms (according to the structure contained within PDB 2F49) which is found to be sufficiently cohesive by our algorithm (note that a distance of 5.8 angstroms easily fits into a ball with a radius smaller than 4 angstrom).

4.4 Peptidase Proteins

A set of 2558 proteins with peptidase activity makes up the final dataset for this analysis. These proteins catalyse a reaction to break up the covalent bonds between peptides. Many of these proteins are therefore involved in the degradation of cellular proteins. There is a great deal of variety in the molecular structure of these proteins as many types of enzymes display peptidase activity. Using a support threshold of 80%, *max_rad* equal to 4.5 angstrom, *min_size* equal to 3, and *max_size* unlimited, a total of 146 patterns were discovered in this dataset and each of these consists of three amino acids. However, in contrast to the previous analyses, the patterns mostly concern amino acids in unstructured regions of the proteins. This is not unsurprising as α-helices are not as prevalent in peptidase proteins as they are in DNA-binding proteins or kinases. Due to the intrinsic diversity of the peptidase dataset, the same patterns are derived from amino acids present in very different domains in different proteins. The most cohesive patterns for this dataset can be found in Table 7 with their respective cohesive radius in angstrom and support.

An example of a peptidase from this dataset, the *E. coli* PepP aminopeptidase in monomer form (as reported by PDB 1A16) is shown in Fig. 7 where the amino acids extracted for the peptidase proteins are provided in colour. The PepP protein is an exopeptidase that cleaves the N-terminal residue from polypeptides. The centre of the protein contains two metal-binding sites, which catalyse the cleavage reaction [26]. Within the PepP protein, five amino acids are known to function as metal-binding residues and two histidine residues are known to be essential for the catalytic activity [27]. Interestingly, several of the peptidase patterns were found in the neighbourhood of the catalytic site. Similar to the findings in the

TABLE 7: The 30 Most Cohesive Patterns From The *Peptidase* Dataset

Itemset	Cohesive radius	Support
GLY_U SER_U ALA_U	3.69	0.91
GLY_U SER_U VAL_U	3.72	0.91
GLY_U SER_U ASP_U	3.72	0.91
GLY_U LEU_U VAL_U	3.76	0.93
GLY_U THR_U ALA_U	3.78	0.93
LEU_U VAL_U ALA_U	3.79	0.93
GLY_U VAL_U THR_U	3.79	0.92
GLY_U SER_U LEU_U	3.80	0.92
GLY_U LEU_U ALA_U	3.80	0.94
GLY_U VAL_U ILE_U	3.82	0.91
GLY_U SER_U THR_U	3.82	0.91
GLY_U SER_U ILE_U	3.83	0.89
GLY_U THR_U ILE_U	3.83	0.9
GLY_U ALA_U ILE_U	3.84	0.91
LEU_U VAL_U ILE_U	3.86	0.91
GLY_U VAL_U PRO_U	3.87	0.94
GLY_U SER_U ASN_U	3.87	0.9
SER_U LEU_U VAL_U	3.88	0.91
SER_U LEU_U ILE_U	3.92	0.89
LEU_U ALA_U ILE_U	3.92	0.91
GLY_U THR_U PRO_U	3.92	0.93
SER_U LEU_U ALA_U	3.95	0.91
GLY_U SER_U TYR_U	3.96	0.89
GLY_U VAL_U ALA_U	3.97	0.93
GLY_U LEU_U ILE_U	3.98	0.91
SER_U LEU_U THR_U	3.99	0.91
GLY_U THR_U ASN_U	3.99	0.92
VAL_U THR_U ILE_U	4.00	0.90
GLY_U ASP_U LYS_U	4.01	0.9
SER_U VAL_U ILE_U	4.01	0.89

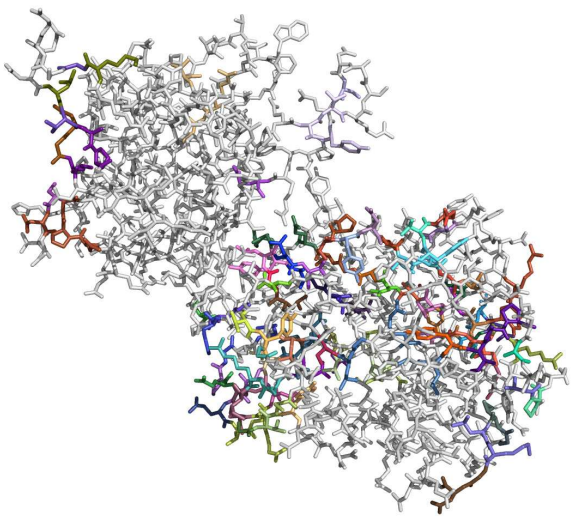

Fig. 7: The Molecular Structure of the PepP Protein

previous analysis, the patterns do not always contain the known functional residues themselves but instead match the amino acids that make up the strand carrying the residue. This indicates that the cohesive patterns do not consist of the amino acids that provide the target specificity but instead correspond to the common residues that stabilise their location. Indeed, several itemsets are found to span different strands that form the metal-binding region. For example, the amino acids within the rule SER_U ALA_U GLY_U (i.e., Serine, Alanine and Glycine in unstructured regions) match residues 228, 269 and 270 respectively. This is a distance of more than 40 residues within the sequence, but the protein folding has brought the αC of these residues to within 5 angstroms. Indeed both these strands form a loop along the centre of the metal-binding site. Furthermore, the strand containing ALA_{269} and GLY_{270} also contains the metal-binding residue ASP_{272}.

5 Conclusions

In this paper, we have presented a novel method with two variations (*VertexOne* and *VertexAll*) to mine frequent cohesive itemsets in multidimensional data. Through experimental evaluation, we confirmed that *VertexOne* outperforms *VertexAll* by finding similar interesting itemsets much faster. The algorithm was applied to datasets containing the full atomic coordinates of various proteins. We were able to successfully identify sets of amino acids that frequently occur in close proximity to each other throughout the given proteins. Thorough analysis revealed that the patterns did indeed reflect amino acids that could span distances in the primary sequence of the protein but were brought together through the protein folding. Furthermore, the types of patterns that we found in the current setting mostly seem to reflect amino acids with a supporting role to the overall or specific structure of the protein.

Acknowledgments

Cheng Zhou is financially supported by the China Scholarship Council (CSC). This work was supported by the Fund for Scientific Research - Flanders (FWO-Vlaanderen) project "Evolving graph patterns".

References

[1] C. Zhou, P. Meysman, B. Cule, K. Laukens, and B. Goethals, "Mining spatially cohesive itemsets in protein molecular structures," in *Proceedings of the 12th International Workshop on Data Mining in Bioinformatics*. ACM, 2013, pp. 42–50.

[2] B. Cule, B. Goethals, and C. Robardet, "A new constraint for mining sets in sequences," in *SDM'09*, 2009, pp. 317–328.

[3] A. Kouranov, L. Xie, J. de la Cruz, L. Chen, J. Westbrook, P. E. Bourne, and H. M. Berman, "The RCSB PDB information portal for structural genomics." *Nucleic acids research*, vol. 34, no. Database issue, pp. D302–5, Jan. 2006.

[4] G. J. Kleywegt, "Recognition of spatial motifs in protein structures." *Journal of molecular biology*, vol. 285, no. 4, pp. 1887–97, Jan. 1999.

[5] J. Huan, W. Wang, A. Washington, J. Prins, R. Shah, and A. Tropsha, "Accurate classification of protein structural families using coherent subgraph analysis," in *Proceedings of the Ninth Pacific Symposium on Biocomputing (PSB)*, 2003, pp. 411–422.

[6] Z.-P. Liu, L.-Y. Wu, Y. Wang, X.-S. Zhang, and L. Chen, "Bridging protein local structures and protein functions." *Amino acids*, vol. 35, no. 3, pp. 627–50, Oct. 2008.

[7] M. Vendruscolo, E. Kussell, and E. Domany, "Recovery of protein structure from contact maps," *Folding and Design*, vol. 2, no. 5, pp. 295–306, Oct. 1997.
[8] J. Hu, X. Shen, Y. Shao, C. Bystroff, and M. J. Zaki, "Mining protein contact maps," in *2nd BIOKDD workshop on data mining in bioinformatics.*, 2002.
[9] J. Huan, W. Wang, D. Bandyopadhyay, J. Snoeyink, J. Prins, and A. Tropsha, "Mining protein family specific residue packing patterns from protein structure graphs," in *Proceedings of the eighth annual international conference on Research in computational molecular biology*. ACM, 2004, pp. 308–315.
[10] J. Huan, D. Bandyopadhyay, J. Prins, J. Snoeyink, A. Tropsha, and W. Wang, "Distance-based identification of structure motifs in proteins using constrained frequent subgraph mining." *Proceedings of the IEEE Computational Systems Bioinformatics Conference*, pp. 227–38, Jan. 2006.
[11] O. Rahat, U. Alon, Y. Levy, and G. Schreiber, "Understanding hydrogen-bond patterns in proteins using network motifs," *Bioinformatics*, vol. 25, no. 22, pp. 2921–2928, 2009.
[12] W. Dhifli, R. Saidi, and E. Mephu Nguifo, "Smoothing 3d protein structure motifs through graph mining and amino acid similarities," *Journal of Computational Biology*, 2013.
[13] B. Gärtner, "Fast and robust smallest enclosing balls," in *Algorithms-ESA'99*. Springer, 1999, pp. 325–338.
[14] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *VLDB'94*. Morgan Kaufmann Publishers, 1994, pp. 487–499.
[15] A. Andreeva, D. Howorth, J.-M. Chandonia, S. E. Brenner, T. J. P. Hubbard, C. Chothia, and A. G. Murzin, "Data growth and its impact on the SCOP database: new developments." *Nucleic acids research*, vol. 36, no. Database issue, pp. D419–25, Jan. 2008.
[16] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock, "Gene ontology: tool for the unification of biology. The Gene Ontology Consortium." *Nature genetics*, vol. 25, no. 1, pp. 25–9, May 2000.
[17] C. E. Bell, P. Frescura, A. Hochschild, and M. Lewis, "Crystal structure of the lambda repressor C-terminal domain provides a model for cooperative operator binding." *Cell*, vol. 101, no. 7, pp. 801–11, Jun. 2000.
[18] C. G. Kalodimos, R. Boelens, and R. Kaptein, "Toward an integrated model of protein-DNA recognition as inferred from NMR studies on the Lac repressor system." *Chemical reviews*, vol. 104, no. 8, pp. 3567–86, Aug. 2004.
[19] D. N. Arvidson, F. Lu, C. Faber, H. Zalkin, and R. G. Brennan, "The structure of PurR mutant L54M shows an alternative route to DNA kinking." *Nature structural biology*, vol. 5, no. 6, pp. 436–41, Jun. 1998.
[20] P. Meysman, K. Marchal, and K. Engelen, "Identifying common structural DNA properties in transcription factor binding site sets of the LacI-GalR family," *Current bioinformatics*, vol. 8, no. 4, 2013.
[21] K. S. Gajiwala and S. K. Burley, "Winged helix proteins," *Current Opinion in Structural Biology*, vol. 10, no. 1, pp. 110–116, Feb. 2000.
[22] H. Sharma, S. Yu, J. Kong, J. Wang, and T. A. Steitz, "Structure of apo-CAP reveals that large conformational changes are necessary for DNA binding." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 106, no. 39, pp. 16 604–9, Sep. 2009.
[23] A. Nakamura, C. Wada, and K. Miki, "Structural basis for regulation of bifunctional roles in replication initiator protein." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 47, pp. 18 484–9, Nov. 2007.
[24] E. D. Scheeff and P. E. Bourne, "Structural evolution of the protein kinase-like superfamily." *PLoS computational biology*, vol. 1, no. 5, p. e49, Oct. 2005.
[25] A. Reményi, M. C. Good, R. P. Bhattacharyya, and W. A. Lim, "The role of docking interactions in mediating signaling input, output, and discrimination in the yeast MAPK network." *Molecular cell*, vol. 20, no. 6, pp. 951–62, Dec. 2005.
[26] W. T. Lowther and B. W. Matthews, "Metalloaminopeptidases: Common Functional Themes in Disparate Structural Surroundings," *Chemical Reviews*, vol. 102, no. 12, pp. 4581–4608, Dec. 2002.
[27] S. C. Graham, P. E. Lilley, M. Lee, P. M. Schaeffer, A. V. Kralicek, N. E. Dixon, and J. M. Guss, "Kinetic and crystallographic analysis of mutant Escherichia coli aminopeptidase P: insights into substrate recognition and the mechanism of catalysis." *Biochemistry*, vol. 45, no. 3, pp. 964–75, Jan. 2006.

Cheng Zhou received the master degree in management of information systems from the National University of Defense Technology, China, in 2011. He is currently working towards a PhD degree in the Department of Mathematics and Computer Science, University of Antwerp, Belgium. His research interests include data mining and its applications.

Pieter Meysman received the BS, MS and PhD degree in bioscience engineering from the KU Leuven, Belgium. He is currently a post-doctoral researcher at the Advanced Database Research and Modeling (ADReM) research group of the University of Antwerp and the Biomedical Informatics Research Centre Antwerp of the University of Antwerp and the University Hospital of Antwerp. He is part of the Student Council of the International Society for Computational Biology and founded its Belgian branch (RSG-Belgium) in 2011. His research interests are mainly focused on data mining algorithms for pattern discovery in structural and evolutionary genomics.

Boris Cule is a post-doctoral researcher at the Department of Mathematics and Computer Science, University of Antwerp, Belgium. He obtained his Ph.D. in Computer Science degree at the same institution in 2012, following the Master in Mathematics degree obtained in 2007. His very first paper introduced the cohesion measure for evaluating the quality of itemsets, which has since been successfully applied to various problem settings in a wide range of domains.

Kris Laukens is coordinator of the biomedical informatics research center and professor at the University of Antwerp. His current research lies in the application of data mining to complex life science data. In 1999 he received a master degree in biology, and in 2003 he obtained his PhD degree based on research work in the field of proteomics, at the University of Antwerp.

PLACE PHOTO HERE

Bart Goethals is professor at the Department of Mathematics and Computer Science of the University of Antwerp in Belgium. He leads the Data Mining lab of the Advanced Database Research and Modeling (ADReM) research group, which performs fundamental research on the structures, the basic properties and the power of languages, algorithms and methodologies for processing and analysing large quantities of data. His primary research interests are the study of data mining techniques to efficiently find interesting patterns and properties in large databases. He received the IEEE ICDM 2001 Best Paper Award and the PKDD 2002 Best Paper Award for his theoretical studies on frequent itemset mining. He was organizer and program chair of ECML PKDD 2008, program chair of SIAM DM 2010, and general chair of IEEE ICDM 2012. He organised and chaired several workshops, such as ICDM FIMI 2003, 2004, PKDD KDID 2004, SIGKDD OSDM 2005, SDM HPDM 2006, and SIGKDD UP 2010 and served on the organizing and program committees of several conferences such as ACM SIGKDD, IEEE ICDM, SIAM DM, and ECML PKDD. He is general chair of the ECML PKDD Steering Committee (2008-2011), associate editor of the Data Mining and Knowledge Discovery journal, the Knowledge and Information Systems journal and Editor-in-Chief of the ACM SIGKDD Explorations newsletter.