# CCMS: A Greedy Approach to Motif Extraction

Giacomo Drago, Marco Ferretti, and Mirto Musci

University of Pavia, Via Ferrata 1, 27100 Pavia, Italy
marco.ferretti@unipv.it,
{mirto.musci01,giacomo.drago01}@ateneopv.it

**Abstract.** Efficient and precise motif extraction is a central problem in the study of proteins functions and structures. This paper presents an efficient new geometric approach to the problem, based on the General Hough Transform. The approach is both an extension and a variation of the Secondary Structure Co-Occurrences algorithm by Cantoni et al. [1-2]. The goal is to provide an effective and efficient implementation, suitable for HPC. The most significant contribution of this paper is the introduction of a heuristic greedy variant of the algorithm, which is able to reduce computational time by two orders of magnitude. A secondary effect of the new version is the capability to cope with uncertainty in the geometric description of the secondary structures.

**Keywords:** Motif Extraction, Secondary Structures, SSC, Hough Transform, Algorithm Optimization, Greedy Algorithm.

## 1 Introduction

The study of proteins is a crucial aspect of the biological field, as they are essential elements of every living cell. Often, the function of a given protein is tightly tied to its geometric structure. Thus, protein structure analysis is an important issue with multiple applications. For example, the ability of a protein to bind other proteins or ligands and the estimation of evolutionary distances between families of proteins are both based on their spatial structure. A key part in the geometric description of a protein is played by the structural motif, a 3D group of secondary structures (SS) which appears in a variety of molecules. Thus, precise and efficient motif identification or extraction is a much requested application in the biological community.

The simplest approach to motif extraction is the entire motif search (EMS). It is used to identify the location of each possible motif (defined by a set of properties, e.g. its spatial dimension) in a given protein. The EMS has been thoroughly analyzed [3], with special regard to its parallel implementation.

However, the most interesting approach to the problem is one that we can call "*cross motif search*" (CMS). In this case every possible motif in a given source protein, is searched in each protein of a given set. A CMS run is composed of multiple iterations: defining the cardinality of a motif as the number of SS in that motif, the $n$-th iteration of a CMS run tries to extract, from the protein set, all the motifs with cardinality $n$.

A. Petrosino, L. Maddalena, P. Pala (Eds.): ICIAP 2013 Workshops, LNCS 8158, pp. 363–371, 2013.

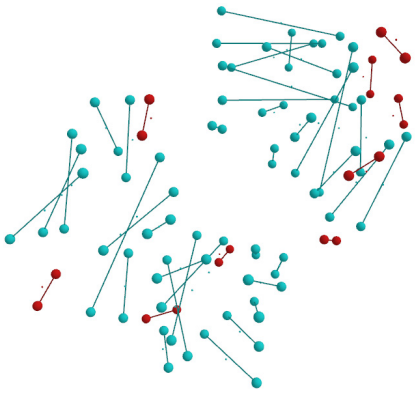

**Fig. 1.** Protein 7fab displayed as a cloud of segments using our ad-hoc visualizer software. *Segments*: SSs. *Spheres*: endpoints. *Dots*: barycenter. *Red*: alpha helices; *Blue*: beta sheets.

In the most generalized variant, a complete CMS (CCMS) searches for every possible motif of every source protein in any other protein in the set.

Of course, as the entire Protein Data Bank [4] contains about 90,000 proteins, a CCMS may require an extremely high computational time.

In this paper we will present the CCMS algorithm and show an efficient implementation, suitable for HPC. The algorithm is based on an existing approach, the Secondary Structure Co-Occurrences (SSC) [1-2], which is itself based on the General Hough Transform technique [5]. The key idea is to ignore the biological significance on the motifs as much as possible, and to focus on the geometric description of the structures which could be simply viewed as vectors in a 3D space.

First, we briefly describe the original SSC algorithm (sec. 2). Then we discuss an extensive set of modifications to the original algorithm, including an efficient greedy variant which is able to reduce the computational times down by two orders of magnitude (sec. 3). Finally, we present a few benchmarks (sec. 4).

## 2 The Base SSC Algorithm

Given a set of oriented segments in a 3D space (the source), and a search space consisting of a certain number of oriented segments, the SSC algorithm [1-2] is able to determine whether the source is present in the search space or not, independently of translations and rotations of the source.

The key idea underlying the algorithm is that any relevant secondary structure (SS) in a protein can be easily modeled as an oriented segment. A segment is obtained from a least-square approximation of all the relevant amino-acid positions, as defined in the DSSP description of a SS [6]. Every segment is simply identified by a barycenter, two endpoints and an orientation (fig. 1). Thus a DSSP file describing an entire protein can be transformed into a list of segments and their properties, and easily described by a simple XML file.

Thanks to this simplified model, the SSC algorithm is able to determine whether a given motif of SSs is present in a given protein or not. Of course, the SSC algorithm can be used as the base element of both the EMS and the CMS algorithms.

### 2.1 Using the GHT to Extract Motifs

The Generalized Hough Transform describes a *geometric pattern* by selecting a *reference point* (RP) in a coordinate system local to the pattern and a set of *features elements* (points, segments, etc.) that make up the pattern and that are geometrically referenced to this point. This structural description is encoded in a *reference table* (RT in the following).

The transform (*voting rule*) consists of producing, for each feature element in the *search space* (the object under scrutiny), a number of candidates for the RP of the geometric pattern; these candidates are collected in a *voting space* where evidence of occurrences of candidates are accumulated.

An analysis of the *voting space* looking for peaks of accumulated occurrences allows identifying places where there exist instances of the looked-for geometric pattern.

This method is often used in pattern recognition. Its computational complexity can be quite relevant, since it depends on the numbers of feature elements used to describe the geometric pattern (i.e., the cardinality of the RT), on the number of feature elements present in the search space, and on the resolution at which the voting space is quantized.

SSC is based on a modified GHT, which searches for motifs (*geometric patterns*) of SSs (*feature elements*) inside a given protein (*search space*). In the SSC approach, the basic feature elements are couples (or co-occurrences) of SSs, instead of single SSs as it would be expected. In this way it is possible to be both orientation- and translation-agnostic during the search.

The voting rule is defined using only the geometric features of the SSs vectors. In the case of SSC, it is a change-of-basis matrix.

Each row of a motif RT consists of a set of relevant geometric parameters for a single couple of SSs taken from that motif, and the relative transformation matrix.

A given feature element can meaningfully produce a vote (using the matrix) only if its geometric parameters *match* against the parameters contained inside at least one row of the RT. In general, this *matching phase* can be used to reduce the number of votes, thus reducing both the possibility of false positives and the execution time. The idea is to identify a certain number of properties that the feature elements in both source and search space must possess and check for them before voting.

In ideal conditions (without false positives and spurious votes), if a motif is present in the protein, the voting process produces a peak of votes with intensity exactly equal to the number of rows of the reference table or $\binom{c}{2}$, where $c$ is the motif cardinality.

A step-by-step analysis of the SSC algorithm is outside the scope of this paper. For a more detailed discussion, see [1-2].

# 3 Extending the SSC: CCMS

## 3.1 Optimizations

As the goal of this paper is to produce an efficient implementation of the CMS algorithm, several optimizations have been introduced into the original SSC algorithm. Here we discuss only the most significant ones.

In a naïve implementation, geometric parameters associated to each couple of SSs are calculated *just-in-time*, i.e. each time that a couple is considered. Therefore, some computations are repeated uselessly multiple times. It is possible to save a significant amount of time if one stores the geometric parameters associated to each couple in a special purpose cache. The cache is filled in completely, for each protein, before the actual search on that protein.

It is also possible to linearize the cache, so that the algorithm reads it in sequence, thus ensuring both space and time locality. The key idea is to associate a set of contiguous natural number to the couples, and is based on the combinatorial number system by D. Knuth [7].

## 3.2 Tolerance Parameters

To adapt SSC for CMS, we need to make it account for tolerance in the geometric features. The first three features to consider are the ones used to construct the reference table, namely the barycenter distance, the line distance and the angle between the two SSs in the couple.

Moreover, it is extremely important to allow for adequate *peak tolerance* during the finding phase of the algorithm, as even extremely similar motifs can produce votes in a spread of a few angstroms. Thus, CMS employs an exact procedure to extract accumulation of votes from the search space, based on the *MaxCliqueDyn* algorithm by Konc et al. [8].

We also implemented two additional checks during the *matching* (see sec. 2.1) between a row in the RT and a couple of SSs in the search space. The first check is biological in nature: two geometrical affine couples can match only if both have the same biological types (e.g. two alpha helices can match only with two other alpha helices). The second check is based on the size of the SSs. Two couples can match only if they are comparable in size.

Accounting for tolerance during the matching phase yields two kinds of different false positives (i.e. accumulations of voting points which cannot be considered valid search hits):

1. Random accumulations of voting points, cast from completely unrelated couples;
2. Accumulations of voting points that are only due to a portion of the searched motif.

Note that random accumulations become more frequent as the number of cast votes increase.

In order to reduce the impact of this problem, one can eliminate all the clearly invalid accumulations points. Thus, peaks with cardinality lower than expected[1] and peaks which are too far away from their predicted reference points are filtered away.

Additionally, a third kind of spurious result is possible: an accumulation of votes referring to a valid motif, plus one or more spurious secondary structures. However, results of this kind cannot be filtered as they may contain peaks which were otherwise not considered.

### 3.3 The Orientation Problem

The base SSC algorithm defines the voting rule for each couple as a change-of-basis matrix (see sec.1), which transforms the coordinates of the motif reference point from the source protein reference system to a reference system which depends only on the geometric properties of that couple. As the SSs are modeled as oriented segments, and each couple defines an internal order of SSs, it is possible to define a procedure to identify a unique reference system for each couple. For more details, see again [1].

For performance reasons, it is not affordable to consider inverse couples (i.e. couple which are made of the same SSs, but taken in the inverse order) as different couples. This would double both the number of comparisons and the size of the cache. Moreover, we noticed that the orientations of the SS segments are dependent on their DSSP description, which can differ even for very similar motifs, leading to geometrically valid matches being discarded due to their opposite orientation.

Thus, an orientation-agnostic approach is needed to ensure correctness, i.e. to avoid losing potential matches.

Given two segments in a 3D space, there are at least four different ways to construct a local reference system, which is completely independent of the external context. Each of them depends on the orientations of the segments and their relative order. Thus we modified the way in which the RT is constructed: each row carries four different voting rules, i.e. four different matrices. This means that each match produces four votes, not one. In this way the core of the algorithm remains the same, it is still possible to exploit linear accesses to the cache and correctness is ensured. The increased cost, of course, is due to the extra time needed to process a single vote and the fact that the number of votes increases by a factor of four. Fortunately the filters help in greatly reducing the number of votes.

To avoid useless recalculation, all the inverse matrices are also stored in a cache, initially empty. Each entry is written only when needed, using appropriate synchronization. In this way the application would be safe from race conditions if it were to be run in parallel.

### 3.4 An Efficient Greedy Variant (Greedy-CCMS)

The CCMS algorithm can be implemented in a straightforward way, which leads to a *brute-force* approach. Each iteration (with increasing motif cardinality, see sec. 1) is

[1] i.e. $\binom{c}{2}$, where $c$ is the motif cardinality. See sec. 2.1.

performed independently from one another and has a complexity of $O\left(\frac{N^c}{c!}\right)$ where $N$ is the number of SSs in the search protein and $c$ is the cardinality of the motif [3].

Fortunately, it is possible to greatly reduce the *average complexity* of the CCMS algorithm. Note, however, that the asymptotic complexity remains the same.

Given two motifs $S$ and $M$, we say that $S$ is a sub-motif of $M$ if the SSs in $S$ are a proper subset of the SSs in $M$. The key idea is that a motif of dimension $n$ is present in the search space only if all its sub-motifs of dimension $n-1$ are also present. Starting from this observation it is possible to define a constructive algorithm:

1. The first step consists of a fast brute-force iteration, where motifs of the source protein having a cardinality of three[2] are searched inside the search protein.
2. All found peaks, before being filtered (see sec. 3.2), are collected in a set (called *unfiltered*).
3. The peaks are filtered and saved in another set (*results*), as they are indeed valid.
4. From the next iteration onward, the algorithm constructs new source motifs in the following way: for each peak in the *unfiltered* set *of the previous iteration*, and for each SS $s$ in the source protein, add $s$ to the motive associated to the peak.
5. Search the so-constructed source motifs inside the search protein.
6. The algorithm terminates when the *unfiltered* set of the previous iteration is empty.

There are two main advantages related to the greedy approach. The first is the extremely lower execution times (up to two orders of magnitude) if compared to the brute-force approach (see sec. 4). The second one is that the approach is feasible even for motifs having a high cardinality, as opposed to the brute-force approach. In fact, Greedy-CCMS can efficiently extract motifs up to the maximum cardinality $N$, even identifying whole tertiary structures.

## 4 Benchmarks

In this section we will present some benchmarks for both the brute-force and the greedy variant of the CCMS.

As stated in sec. 1, a CCMS on a protein set implies comparing each protein in the set against each other, in order to extract all the common motifs.

The small protein dataset described in [2] has been used for the benchmarks. As the dataset contains 20 proteins, the total number of protein pairs to be processed is exactly $20^2 - 20 = 380$. The maximum cardinality for the source motifs was set to 5.

The results reported in Table 1 clearly show the performance improvement of the greedy variant of the algorithm over the brute-force one. The speed-up of the greedy variant is about 75.6 using strict tolerance parameters and 42.3 using loose parameters. Both implementations take advantage from the optimizations described in sec. 3. For higher cardinalities of the source motifs the speedup of the greedy variant is expected to be even better.

[2] The smallest significant dimension for a motif is three, as dimension one is nonsense, and dimension two is trivial.

**Table 1.** CCMS run on the test database (max. cardinality 5): execution times

| CCMS variant | Tolerance | Total exec. time (sec.) | Avg. exec. time for a single pair of proteins (sec.) |
|---|---|---|---|
| brute-force | strict[3] | 2,756.84 | 7.255 |
| brute-force | loose[4] | 2,821.06 | 7.424 |
| greedy | strict | 36.51 | 0.096 |
| greedy | loose | 66.62 | 0.175 |

It is worth mentioning that as the tolerance parameters become looser, the greedy variant tends to the same performance of the brute-force variant. The reason is that the higher number of matches for motifs of low cardinality (3 and 4) increases the number of motifs to be processed in the steps 4 and 5 of the Greedy-CCMS (see sec. 3.4).

Exploiting the high performance of the greedy variant, we repeated the benchmark, setting the maximum cardinality to 10. The number of matches, even on a very small dataset, is quite impressive (see Table 2) [5]. Fig. 2 depicts an example of match with cardinality 10.

**Table 2.** CCMS run on the test database (max. cardinality 10): histogram of the matches

| Tolerance | Total exec. time (sec.) | Motif cardinality | Number of matches |
|---|---|---|---|
| strict | 47.25 | 3-4 | 4,551 |
| | | 5-8 | 2,313 |
| | | 9-10 | 0 |
| loose | 119.33 | 3-4 | 5,007 |
| | | 5-8 | 3,966 |
| | | 9-10 | 611 |

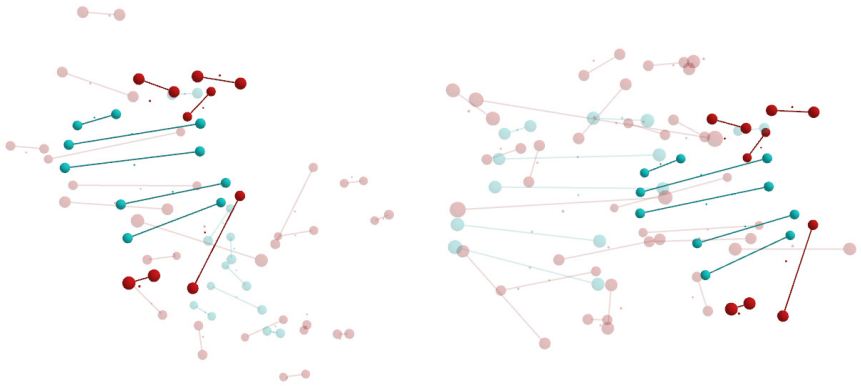

**Fig. 2.** CMS between proteins 2qx8 (*source*) and 2qx9 (*search*). Example of match with cardinality 10.

[3] Barycenter distance: 0.5 Å; line distance: 0.5 Å; angle: 5°; peak: **1.0** Å.

[4] Barycenter distance: 0.5 Å; line distance: 0.5 Å; angle: 5°; peak: **1.5** Å.

[5] Numbers may include the same motifs matching more than one time. Moreover, items like 2qx8 and 2qx9 represent variants of the same protein, thus producing a large number of matches.

## 5 Conclusions and Future Work

In this paper we have presented a new algorithm, called CCMS, which is based on an extension of the pre-existent SSC algorithm. Benchmarks show that Greedy-CCMS is extremely more efficient than the straightforward brute-force implementation, so that the CCMS is indeed feasible in terms of execution times.

The number of extracted motifs is impressive. However, the current implementation is unable to automatically exclude duplicate results, to discriminate the most relevant results from a biological point of view, or to perform a statistic analysis of the matches. Addition of efficient data-mining is the main line of work for the next extension of the algorithm.

Moreover, it may be interesting to analyze a different variant of the original algorithm, by the same authors, called SS Tern [9]. The base feature element for the SST is a triplet of SSs, and not a couple as in the SSC. Using terns, it is easier to unambiguously identify a local reference system, thus decreasing the size of a single row in the reference table, even if the SST has a higher complexity.

Eventually it could be possible to implement a new CMS variant using as its base elements modified versions of both SSC and SST.

Finally, as it was noted previously, CCMS can be very computationally expensive, even in the greedy variant. An MPI implementation of the algorithm is in the workings; the goal is to process the entire PDB (or at least the most relevant SCOP fields [10]) on the CINECA supercomputing facilities [11].

## References

1. Cantoni, V., Ferone, A., Ozbudak, O., Petrosino, A.: Structural analysis of protein secondary structure by GHT. In: 21st International Conference on Pattern Recognition, ICPR 2012, Tsukuba, Japan, November 11-15, pp. 1767–1770. IEEE Computer Society Press (2012)
2. Cantoni, V., Ferone, A., Ozbudak, O., Petrosino, A.: Motif Retrieval by Exhaustive Matching and Couple Co-occurrences. In: 9th International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics, CIBB 2012, Texas, July 12-14 (2012)
3. Ferretti, M., Musci, M.: Entire Motifs Search of Secondary Structures in Proteins: A Parallelization Study. In: International Workshop on Parallelism in Bioinformatics EUROMPI 2013, Madrid, Spain, September 17 (in printing, 2013)
4. Protein Data Bank, `http://www.rcsb.org/pdb`
5. Ballard, D.: Generalizing the Hough Transform to Detect Arbitrary Shapes. Pattern Recognition 13(2), 111–122 (1981)
6. Kabsch, W., Sander, C.: Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. Biopolymers 22, 2577–2637 (1983)
7. Knuth, D.E.: Generating All Combinations and Partitions. In: The Art of Computer Programming, vol. 4, Fascicle 3, pp. 5–6. Addison-Wesley (2005)
8. Konc, J., Janežič, D.: An improved branch and bound algorithm for the maximum clique problem. MATCH Communications in Mathematical and in Computer Chemistry 58(3), 569–590 (2007)

9. Cantoni, V., Ferone, A., Ozbudak, O., Petrosino, A.: Protein motifs retrieval by SS terns occurrences. Pattern Recognition Letters 34, 559–563 (2012)
10. Structural Classification of Proteins and ASTRAL (January 2013), http://scop.berkeley.edu
11. CINECA supercomputing center (9th in top500.org as of May 2013), http://www.cineca.it