Protein Annotation by Secondary Structure Based Alignments (PASSTA)

Constantin Bannert[1] and Jens Stoye[1]

Technische Fakultät, Universität Bielefeld, Germany.

Abstract. Most software tools in homology recognition on proteins answer only a few specific questions, often leaving not much room for the interpretation of the results. We develop a software *Passta* that helps to decide whether a protein sequence is related to a protein with known structure. Our approach may indicate rearrangements and duplications, and it displays information from different sources in an integrated fashion.

Our approach is to first break each sequence of the Protein Data Bank (PDB) into Secondary Structure Elements (SSEs). Given a query sequence, our goal is then to 'explain' it by SSE sequences as good as possible. Therefore, we use the Waterman-Eggert algorithm to compute pairwise alignments of SSE sequences with the query. In a graph-based approach, we then select those alignments that reproduce the query in an optimal way. We discuss two examples to illustrate the potential (and possible pitfalls) of the method.

1 Introduction

The need to characterize and annotate the enormous amount of gene sequences emerging from the genome sequencing projects led to the development of many useful algorithms and tools. A common approach used is homology recognition, where a query sequence is compared to one or many already characterized sequences or structures. If a certain similarity between those can be found, we can assume the existence of a common ancestor, and hence, homology. Shi *et al.* [18] defined four major groups of homology recognition:

1. Methods that do pairwise sequence comparison, usually by computing pairwise alignments. They are able to detect closely related homologs, but often miss remote homologies.
2. Tools in the second group are also based on sequence comparison. However, they use multiple alignments of related sequences and compute profiles or probabilistic models from them to improve the detection of remote homologs.
3. The third group of methods uses structure- and sequence information.
4. Homology detection in the fourth group relies on structure information only. Methods in this group are usually threading methods.

BLAST [1], FASTA [16], and the Smith-Waterman algorithm [20] are well known examples from the first group of methods. They compute pairwise alignments, but are usually used to search a whole sequence database.

PSI-Blast [2] can be seen as an enhanced BLAST. It is better suited to detect remote homologies, and since it computes and uses profiles of simililar sequences, it can be assigned to the second group. The same holds for HMM-based approaches, see [9] and references in [18].

The Jumping Alignment algorithm 'Jali' [21] also belongs to the second group of methods, but the concept already suggests a connection to protein structure. The query is aligned simultaneously to all sequences in a multiple alignment (usually derived from a protein family). However, only one sequence in the alignment, the 'reference sequence', contributes to the actual computation of the Jali score. The algorithm may change (or 'jump' between) the sequences, if the properties of the multiple alignment allow to do so. Structure may play a role, if the aligned family has modular properties, e.g., divides into two subfamilies. However, tests in [3] whether the jumps of the algorithm reflected the secondary structure of some protein families revealed only few examples where this was the case.

All approaches mentioned so far are strictly sequence-based alignment approaches. Since they process their information sequentially, they are not well suited for the detection of rearrangements and duplications. When used in a database search context, the results are usually presented as alignments of the query to the corresponding database hits. If the hit is only partial, it is not at once clear whether the unmatched part of the query bears similarities to other proteins in the database.

This was our motivation to develop a new software, called *Passta* (Protein annotation by secondary structure based alignments). Passta is a fragment-based alignment approach on secondary structure elements (SSEs) or, more precisely, SSE sequences. SSEs can be seen as the smallest structural entities in a protein, and we decided to use them, even though their structure is not fixed *in vivo*, but depends on the environment and other factors (see, e.g. [5]). Given a query sequence, the aim of Passta is to show how well it can be represented with SSEs found in sequences of the PDB. Further information is provided by linking the SSEs to the SCOP classification database [11, 14], and by showing the position of the matched SSEs in their chain, which helps to find possible rearrangements and duplications. Each site in the query can only be aligned with one SSE at a time, but we display all such alignments simultaneously.

Two methods in the third of the groups listed above also use SSEs, MAP [17] and SEA [23]. Both use predicted SSEs. MAP derives a secondary structure 'map' from the SSEs to find the most likely fold from a database of domains with known structure. SEA ('SEgment Alignment') uses a graph-based approach to compare two protein sequences. For both proteins, SSEs are predicted with several secondary structure prediction methods and represented in two unweighted graphs. Ye *et al.* [23] then solve a network matching problem: They search for a path in each graph/network, such that the corresponding SSEs in both paths are maximally similar.

The main difference between their and our concept is that we represent residue-level alignments from many different proteins simultaneously in one graph and search for those that best explain the query, while Ye *et al.* use one graph for each target. Therefore, the SEA approach can not detect similarities to different database hits at the same time.

Other methods in homology recognition using SSEs are mostly in group four. They use the three-dimensional coordinates of the SSEs, mostly for vector representations in protein structure comparison. These methods are basically out of scope here, however, we would like to mention a recent study by Shih and Hwang [19] who investigated alternative/permuted alignments by structural comparison, where the SSEs were not required to be sequential. Their results indicate that this area is to some degree overlooked, and investigations here will be useful to improve our understanding of the organization and evolution of proteins.

We present the basic framework of Passta in Section 2 and illustrate its performance in Section 3. The first example we give is a plastocyanin sequence from *Oryza sativa* (rice), which serves as a proof of concept. The second one is the CASP6 target 'T0269' (see `http://predictioncenter.llnl.gov/casp6/Casp6.html`).

2 Material and Methods

Terminology: Based on the atomic coordinates of a structure determination experiment, each amino acid in a protein structure is assigned a secondary structure *state* (see e.g., [5], chapter 17). The standard here is the DSSP algorithm [12], which was also used to assign the secondary structure states to the databases in Section 2.1. These states can be grouped into *classes*. The DSSP states G, H, and I are helical states, B and E are strand states, and the remaining three (T, S, and blank) are random coil or loop states. A 'SSE' is a protein segment where all amino acid states are equal or at least belong to the same class.

2.1 Database integration

Passta uses a relational database (Passta DB) that integrates information from three secondary source databases which themselves are all derived from the Protein Data Bank (PDB) [4].

The **Protein Topology Graph Library (PTGL)** [13] is based on the atomic coordinates of PDB proteins satisfying certain quality criteria. These proteins were decomposed into SSEs of known local structure, and their topology was stored. The aim of the PTGL is to provide this topology information to the user, but we currently use only the decomposed SSEs.

The **PDBFinder II database** (submitted) is an enhanced version of the PDBfinder database [10]. It provides extensive information for almost all proteins in the PDB. Most of this information could also be found in other databases as well, but here it is all in one place. We parse loops and coils from the PDBFinder II database.

The **SCOP database** [11, 14] is a classification on protein domains. It defines four hierarchical levels: *family*, *superfamily*, *common fold*, and *class*. Two domains in the same family are closely related, indicated by a high percentage of sequence identity and structural similarity. Domains in the same superfamily are rather distantly related, but should have a common ancestor. Mostly, this means low sequence identity, but high structural similarity. Two domains have a common fold if their secondary structure elements have the same arrangement, i.e. topology. However, they are not necessarily related. The different folds are grouped into classes according to their main class of secondary structure. The ASTRAL database [6, 7] can be seen as bridging the gap between SCOP and the PDB. Here, we use the 'SPACI' score that the ASTRAL consortium assigns to each protein domain in SCOP. It summarizes the quality of the structure determination experiment. We use it to determine a representative among sequence-identical SSEs and Chains.

Integration. After the integration of the source databases into the Passta DB, it contains tables and data for most proteins, chains, and SSEs available. Also, some precomputed information needed in the annotation approach ('PasstaRun', see Section 2.2) was stored.

The PDB and Chain ID fields are common in all of the source databases, so we used them to cross-index all information before storing it in the Passta database.

The SSEs with helical or strand conformation were taken from the PTGL. The coils were parsed from the PDBFinderII, because the PTGL does not provide them. To ensure consistency, we required the SSE sequences from the PTGL to map back to the PDBfinder II chain sequence. If this was not possible, we excluded the whole chain from the database. If more than twenty percent of a SSE sequence was made up of 'X's (amino acid unknown), we excluded this SSE as well. Some SSEs are not contigous, they contain a chain break, indicated by a gap character ('$-$'). We decided to split those SSEs into two of the same class. After the integration, the Passta DB contains 21572 proteins, 44048 chains, and almost 1.5 million SSEs. That is about 90 % of all possible data.

MaxScores. For each SSE in the database, we stored its maximal alignment (i.e., exact match) score under several substitution matrices. We use these values as rough estimates of alignment quality (see Section 2.2).

Redundancy. Many chain- and SSE sequences are not unique. Since redundant sequences slow down the search procedure, we mark them in order to exclude them from the search process. The procedure is basically the same for chain and SSE sequences: We select all equal sequences and compare the SPACI scores of their 'parent' proteins. The sequence with the best associated SPACI score is marked as being the representative, i.e. non-redundant. First, we do this on the level of chains. Then, we apply it to the SSEs from those chains that were marked non-redundant right before.

SCOP. We also integrated the SCOP classification into the database. For most SSEs we now know the structural domain it belongs to.

2.2 PasstaRun - Annotation Strategy

The annotation of the query with PDB SSEs is implemented in two stages. The first stage ('Pass One') is a filtering approach. It selects some candidate chains for use in the second stage ('Pass Two'), the annotation process itself. We used the blosum62 substitution matrix with gap costs of 12 and 2 for initiation and extension, respectively.

Pass One. Given a query R of length n, Pass One starts by selecting all non-redundant SSEs of length 6 or more from the database. Each SSE comes with its associated *MaxScore*. The Waterman-Eggert algorithm [22] computes pairwise, local, non-intersecting alignments of two sequences. It starts with the optimal, i.e. highest scoring alignment, then co- and suboptimal alignments are computed. We apply the Waterman-Eggert algorithm in Pass One to compute several alignments between each SSE sequence and R, until the ratio $score/MaxScore$ drops below a predefined constant.

Let A be the set of all alignments found in this way. Each alignment $\alpha \in A$ is represented by a 5-tuple (b, e, c, p, s). Elements b and e, $1 \leq b \leq e \leq n$, are the begin and end indices of the aligned SSE w.r.t. the query; $c \geq 1$ is a unique chain identifier; $p \geq 1$ is the position of the SSE in its chain; and s is the alignment score. For a given alignment $\alpha = (b, e, c, p, s)$, we will refer to the individual components of the 5-tuple as $b(\alpha) := b$, $e(\alpha) := e$, $c(\alpha) := c$, $p(\alpha) := p$, and $s(\alpha) := s$, respectively.

Our goal in Pass One is to find a set of good candidate chains for use in Pass Two. In fact, we only need to find a set of good alignments, since we know the chains that an aligned SSE sequence is contained in. We use a graph-based approach to solve this problem. We define a directed acyclic graph $G = (V, E)$, where the set of vertices V is made up of representations of all alignments in A, plus two other vertices; $head = (0, 0, 0, 0, 0)$ and $tail = (n+1, n+1, 0, 0, 0)$, such that $V = A \cup \{head, tail\}$.

An edge exists between two vertices $u, w \in V$, $u \neq w$, if and only if

1. u and w do not overlap (and u is before w), i.e. $e(u) < b(w)$, and
2. there is no alignment v between u and w, i.e. $\nexists\, v \in V : e(u) < b(v)$ and $e(v) < b(w)$.

A *path* P in G is a sequence of vertices $(v_1, v_2, \ldots, v_k)$ such that v_i and v_{i+1} are connected by an edge for all $1 \leq i < k$. Any path from *head* to *tail* corresponds to a selection of non-overlapping alignments. The *weight* of a path $P = (v_1, v_2, \ldots, v_k)$ is given by $weight(P) := \sum_{i=1}^{k} s(v_i)$, it indicates the selection quality. So, the problem to find good candidate chains transforms to a *single-source shortest path* problem from *head* to *tail*, which can be solved easily and efficiently (see e.g. [8]).

We compute all such optimal paths, i.e. where $weight(P)$ is maximal, and collect all chain IDs of their vertices for use in Pass Two. However, the SSEs we use in the alignments are all non-redundant. SSEs with an identical sequence may also exist in other chains. We identify those and collect their chain IDs as well.

Pass Two. The goal of the second pass is to annotate the query with the best selection of SSEs from those chains that passed Pass One. Since the result should be biologically feasible, we have placed certain constraints on the algorithm. Otherwise, Pass One and Pass Two are quite similar. We describe changes in the definitions and the algorithm where applicable. Pass Two first collects for each chain c in the list *all* SSEs, regardless of redundancy status or size (i.e., the complete chain). Then we recompute the set of alignments A. However, it makes no sense to align sequences of length one or two to the query. Therefore, we divided the alignment phase into an *align* and an *extend* part.

Align: Let $l(S)$ be the length of an SSE sequence S. If $l(S) = 2$, we compute all exact matches between S and the query R and insert the corresponding alignment(s) into A. If $l(S) \geq 3$, we use the Waterman-Eggert algorithm. We accept an alignment α between S and R whenever its $score/MaxScore$ ratio is larger than a predefined constant.

Extend: Each time we insert an alignment α into A in the *align* phase, we look at the SSEs adjacent to S in its chain. If they exist and their length is less or equal to 4, we align them to R, allowing neither insertions nor deletions. If the score is larger than zero, we include the new alignment into A (for an example, see Fig. 1).

The directed acyclic graph G is built as described in Section 2.2. However, there are some differences: In Pass Two, we expand the edge definition. There is now also an edge between two vertices $u, w \in V$, if:

1. $c(u) = c(w)$ i.e., both SSEs come from the same chain,
2. $e(u) < b(w)$ (the no-overlap condition from Pass One), and
3. there is no other alignment v between u and w, where the SSE in v comes from the same chain as the one in u and w: $\nexists v \in V : c(u) = c(v) = c(w)$ and $e(u) < b(v)$ and $e(v) < b(w)$.

The definition of a *path* is the same as in Pass One. However, a path $P = (v_1, v_2, \ldots, v_k)$ in Pass Two can contain jumps and rearrangements. If the chains of two adjacent vertices in P are different, i.e. $c(v_i) \neq c(v_{i+1})$, we call this a *jump*. If they are equal but their positions are not consecutive, i.e. $c(v_i) = c(v_{i+1})$ and $p(v_{i+1}) - p(v_i) \neq 1$, we call this a *rearrangement*. Let $j(P)$ be the number of jumps in a path P, and $r(P)$ the number of rearrangements. We penalize jumps and rearrangements by two parameters, *jump cost* (jc) and *rearrangement cost* (rc). The weight of a path P is now given by

$$weight(P) \quad = \quad \sum_{i=1}^{k} s(v_i) - j(P) \times jc - r(P) \times rc.$$

This makes the annotation of the query with small chance hits from different chains highly unlikely, if the jump cost is chosen well. Finally, the alignments in the optimal path are visualized in a HTML page.

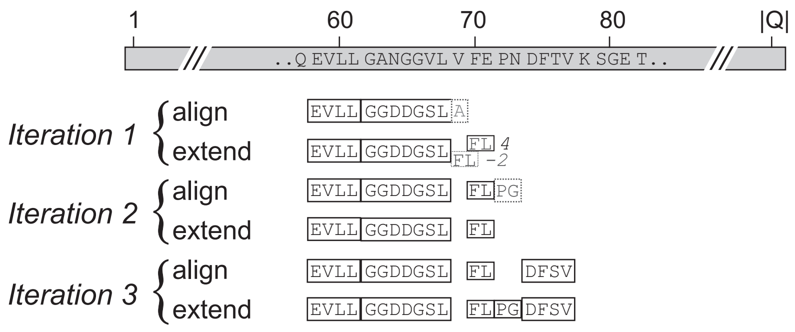


Fig. 1. Example to illustrate the alignment phase in Pass Two. The example corresponds to the result shown in Fig. 2. Alignments displayed with reduced size are from an 'extend' phase. *Iteration 1, align:* A local alignment of the SSE sequence `'GGDDGSLA'` and the query is computed, yielding (`'GGDDGSL'`, `'GANGGVL'`). *Iteration 1, extend:* The SSE left of `'GGDDGSLA'` is already aligned, so no extension is performed here. However, the right SSE is small enough, and we match it (a) at the end position of the **local** alignment and (b) at the position where the **global** alignment would have ended. Only the latter is accepted here because its score exceeds zero. *Iteration 2, align:* 'PG' is of length two, and there is no exact match with the query, so it is not aligned. *Iteration 2, extend:* No alignment was accepted in the alignment phase, therefore nothing is to be extended. *Iteration 3, align:* The SSE sequence `'DFSV'` is aligned to the query. *Iteration 3, extend:* Now it is possible to extend with `'PG'` to the left of `'DFSV'`.

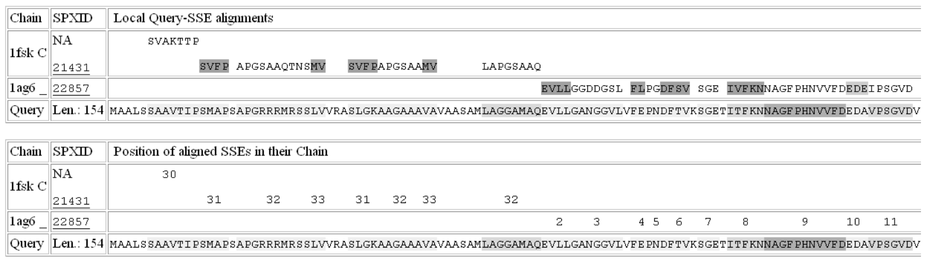
3 Results and Discussion

We present and discuss two examples that we annotated with Passta to illustrate some application possibilities. The complete and colored versions of the presented alignments can be found at `http://www.cebitec.uni-bielefeld.de/~bannert/res/`*filename*, where *filename* is `Pcya36-6.htm` or `T0269-40-5.htm`.

3.1 Plastocyanin from rice

In the first example, the query was a plastocyanin (PC) sequence with a length of 154 amino acids, from rice. We used a jumpcost value of 36. An excerpt of the resulting file is shown in Fig. 2. Passta aligned two PDB chains to the query, the raw score of the optimal path is 369. The annotation suggests that the first part of the alignment is similar to 1fsk_C, the heavy chain of an Immunoglobulin (IG) antibody from *Mus musculus* (mouse). There are two rearrangements (SSEs not consecutive) in the order of the SSEs from this chain. The second chain '1ag6' is a PC, from spinach (*Spinacia oleracea*). The annotation with '1ag6' is doubtlessly correct, however, given the high sequence similarity of the query to other PCs in the PDB, it is not surprising.

What about the IG matched to the first part of the query? It is classified into another SCOP fold than '1ag6', namely 'Immunoglobulin-like beta-sandwich'



Chain	SPXID	Local Query-SSE alignments
1fsk C	NA	SVAKTTP
	21431	SVFP APGSAAQTNSMV SVFPAPGSAAMV LAPGSAAQ
1ag6 _	22857	EVLLGGDDGSL FLPGDFSV SGE IVFKNNAGFPHNVVFDEDEIPSGVD
Query	Len.: 154	MAALSSAAVTIPSMAPSAPGRRRMRSSLVVRASLGKAAGAAAVAVAASAMLAGGAMAQEVLLGANGGVLVFEPNDFTVKSGETITFKNNAGFPHNVVFDEDAVPSGVDV

Chain	SPXID	Position of aligned SSEs in their Chain
1fsk C	NA	30
	21431	31 32 33 31 32 33 32
1ag6 _	22857	2 3 4 5 6 7 8 9 10 11
Query	Len.: 154	MAALSSAAVTIPSMAPSAPGRRRMRSSLVVRASLGKAAGAAAVAVAASAMLAGGAMAQEVLLGANGGVLVFEPNDFTVKSGETITFKNNAGFPHNVVFDEDAVPSGVDV

Fig. 2. Passta alignment of Plastocyanin from rice (excerpt). The upper table shows the local alignments of the SSE sequences with the query. The lower table displays the position of the aligned SSEs in their PDB chain. While the SSEs of '1ag6' are consecutively aligned, those of '1fsk' are to some degree rearranged and duplicated. Some query segments and SSEs are shaded. Dark SSE segments are strand SSEs, light ones are helix SSEs. The intensity of the shaded segments in the query string corresponds to the score they contribute. The darker, the better the score of the segment.

instead of 'Cupredoxin'. The query PC in this experiment is 154 amino acids long. The length of the other PCs in the Passta database is only about 100 residues. A multiple alignment of the query and all non-redundant PCs from the PDB shows that the first 55 residues of the query are not matched by any other PC in the DB (see Fig. 3). An alignment of the query against the whole SCOP family 49504 'Plastocyanin/azurin-like' reveals that the first 20 positions of the query are unmatched by *any* other domain in the family, and that the sequence similarity within the first 50 residues is in general quite low (data not shown). Therefore, we could not find a close homolog matching this region. The IG is not even a remote homolog of the query, but according to Russell *et al.* in [17] it has a loose structural similarity to PCs.

In SCOP, structural similarities are classified into the same fold. The classification depends on the topology of the SSEs. Here, it matters whether a beta-sheet is parallel or antiparallel. The loose similarity observed in our example rather corresponds to the *architecture* level as defined in CATH [15]. The orientation of the SSEs is not important at this level. A loose structural similarity being more informative than a hit with absolutely no relationship to the query, we consider the annotation with the mouse IG as success.

3.2 CASP target T0269

T0269 (PDB code '1vgs') is a thioredoxin peroxidase from the archaeon *Aeropyrum pernix*, with two domains and a length of 250 residues. In this experiment, we used a minimum length of 5 (instead of 6) for the non-redundant SSEs that were aligned in Pass One. The Passta alignment reached a raw score of 191 and used three chains, '1n8j_A', '1prx_A', and '1uth_A' (see Fig. 4). There are eight rearrangements altogether, six in the alignment sequence of '1prx_A' and two in '1uth_A'.

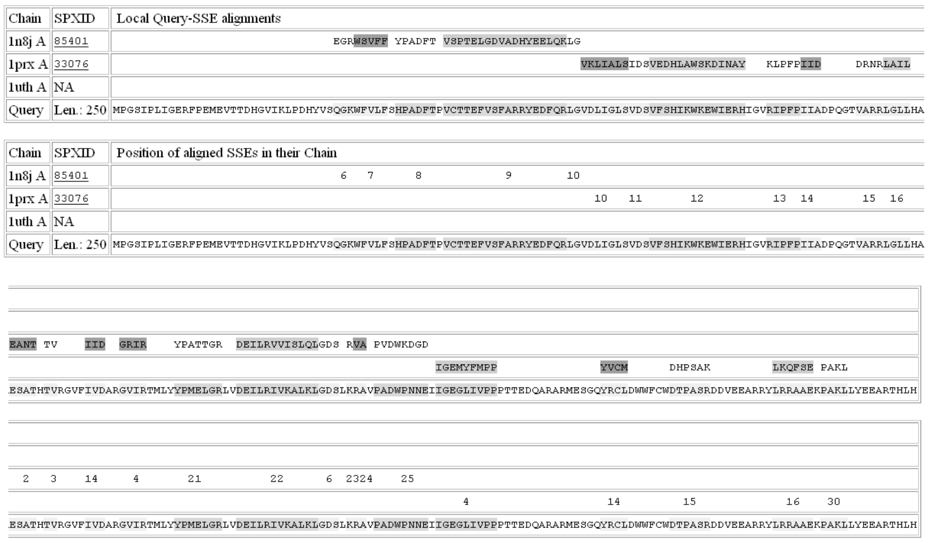
```
PC03            -------------------------------------------------------ETFTV
PC07            -------------------------------------------------------ETYTV
PC08            -------------------------------------------------------ANATV
PC09            -------------------------------------------------------ANATV
PC16            -------------------------------------------------------ANATV
PC04            -------------------------------------------------------QTVAI
PC01            -------------------------------------------------------ASVQI
PC12            --------------------------------------------------------MIDV
PC14            ---------------------------------------------------------IDV
PC17            ---------------------------------------------------------LEV
PC02            ---------------------------------------------------------VEV
PC13            ---------------------------------------------------------VEV
PC05            ---------------------------------------------------------AEV
PC06            ---------------------------------------------------------AEV
PC18            ---------------------------------------------------------AEV
PLAS_ORYSA      MAALSSAAVTIPSMAPSAPGRRRMRSSLVVRASLGKAAGAAAVAVAASAMLAGGAMAQEV
PC19            --------------------------------------------------------DATV
PC10            --------------------------------------------------------AQIV
PC15            --------------------------------------------------------AAIV
PC11            ---------------------------------------------------------AKV
                                                                           :

PC03            KMGADSGLLQFEPANVTVHPGDTVKWVNNKLPPHNILFDDKQVPG-ASKELADKLSHSQ-
PC07            KLGSDKGLLVFEPAKLTIKPGDTVEFLNNKVPPHNVVFDAALNPA-KSADLAKSLSHKQ-
PC08            KMGSDSGALVFEPSTVTIKAGEEVKWVNNKLSPHNIVFAADGV----DADTAAKLSHKG-
PC09            KMGSDSGALVFEPSTVTIKAGEEVKWVNNKLSPHNIVFAADGV----DADTAAKLSHKG-
PC16            KMGSDSGALVFEPSTVTIKAGEEVKWVNNKLSPHNIVFDADGV----PADTAAKLSHKG-
PC04            KMGADNGMLAFEPSTIEIQAGDTVQWVNNKLAPHNVVVEGQ-----------PELSHKD-
PC01            KMGTDKYAPLYEPKALSISAGDTVEFVMNKVGPHNVIFDKVPAG-----ESAPALSNTK-
PC12            LLGADDGSLAFVPSEFSCSPGCKIVFKNNAGFPHNIVFDEDSIP---SGVDASKISMSEE
PC14            LLGADDGSLAFVPSEFSISPGEKIVFKNNAGFPHNIVFDEDSIP---SGVDASKISMSEE
PC17            LLGSGDGSLVFVPSEFSVPSGEKIVFKNNAGFPHNVVFDEDEIP---AGVDAVKISMPEE
PC02            LLGGDDGSLAFLPGDFSVASGEEIVFKNNAGFPHNVVFDEDEIP---SGVDAAKISMSEE
PC13            LLGGDDGSEAFLPGDFSVASGEEIVFKNNAGFPHNVVFDEDEIP---SGVDAAKISMSEE
PC05            LLGSSDGGLAFVPSDLSIASGEKITFKNNAGFPHNDLFDEDEVP---AGVDVTKISMPEE
PC06            LLGSSDGGLAFVPSDLSIASGEKITFKNNAGFPHNDLFDKKEVP---AGVDVTKISMPEE
PC18            KLGSDDGGLVFSPSSFTVAAGEKITFKNNAGFPHNIVFDEDEVP---AGVNAEKISQPE-
PLAS_ORYSA      LLGANGGVLVFEPNDFTVKSGETITFKNNAGFPHNVVFDEDAVP---SGVDVSKISQEE-
PC19            KLGADSGALEFVPKTLTIKSGETVNFVNNAGFPHNIVFDEDAIP---SGVNADAISRDD-
PC10            KLGGDDGSLAFVPSKISVAAGEAIEFVNNAGFPHNIVFDEDAVP---AGVDADAISYDD-
PC15            KLGGDDGSLAFVPNNITVGAGESIEFINNAGFPHNIVFDEDAVP---AGVDADAISAED-
PC11            EVGDEVGNFKFYPDSITVSAGEAVEFTLVGETGHNIVFDIPAGAPGTVASELKAASMDEN
                 :*       : *  .   .*  : :       ** :.                 *
```

Fig. 3. Excerpt of a `ClustalW` alignment of the query ('PLAS_ORYSA') with all other non-redundant plastocyanins in the PDB.

There is no SCOP classification available for '1uth_A', and Fig. 4 shows that the similarities of this chain to the query are few. Its use is probably due to a chance hit of SSE number 4. '1n8j_A' and '1prx_A' are classified into the same SCOP family ('Glutathione peroxidase-like'), which contains thioredoxins. '1n8j_A' is an alkyl hydroperoxide reductase from *Salmonella typhimurium*, '1prx_A' is a human peroxidase. So, both proteins are correctly chosen from the database.

However, we selected this example because of the interesting sequence of SSE alignments in '1prx_A'. There are six rearrangements here, but a closer look reveals that there are really only three sequences of SSEs. If SSE number 14 was removed between number three and four, the first sequence is (10-16), the second one (2-6), and the last one (21-25). Of course, since the alignments



```
Chain   SPXID       Local Query-SSE alignments
1n8j A  85401                                       EGRWSVFF YPADFT VSPTELGDVADHYEELQKLG
1prx A  33076                                                                           VKLIALSIDSVEDHLAWSKDINAY   KLPFPIID     DRNRLAIL
1uth A  NA
Query   Len.: 250   MPGSIPLIGERFPEMEVTTDHGVIKLPDHYVSQGKWFVLFSHPADFTPVCTTEFVSFARRYEDFQRLGVDLIGLSVDSVFSHIKWKEWIERHIGVRIPFPIIADPQGTVARRLGLLHA

Chain   SPXID       Position of aligned SSEs in their Chain
1n8j A  85401                                         6  7      8            9        10
1prx A  33076                                                                             10   11       12          13  14       15  16
1uth A  NA
Query   Len.: 250   MPGSIPLIGERFPEMEVTTDHGVIKLPDHYVSQGKWFVLFSHPADFTPVCTTEFVSFARRYEDFQRLGVDLIGLSVDSVFSHIKWKEWIERHIGVRIPFPIIADPQGTVARRLGLLHA


EANT TV    IID  GRIR    YPATTGR  DEILRVVISLQLGDS RVA PVDWKDGD
                                                              IGEMYFMPP               YVCM      DHPSAK         LKQFSE PAKL
ESATHTVRGVFIVDARGVIRTMLYYPMELGRLVDEILRIVKALKLGDSLKRAVPADWPNNEIIGEGLIVPPPTTEDQARARMESGQYRCLDWWFCWDTPASRDDVEEARRYLRRAAEKPAKLLYEEARTHLH


  2   3    14     4       21          22      6  2324    25
                                                                  4                    14         15             16    30
ESATHTVRGVFIVDARGVIRTMLYYPMELGRLVDEILRIVKALKLGDSLKRAVPADWPNNEIIGEGLIVPPPTTEDQARARMESGQYRCLDWWFCWDTPASRDDVEEARRYLRRAAEKPAKLLYEEARTHLH
```

Fig. 4. Passta alignment of T0269 from *Aeropyrum pernix*. The residues 1-32 are unmatched. Then, five SSEs of '1n8j_A' are consecutively aligned to the query (number 6 to 10). From there on, '1prx_A' is used to annotate the query. After the alignment of SSE number 16, a second sequence SSEs is consecutively aligned, from number 2 to 6. However, the sequence is interrupted by number 14, and after number 4 a third sequence of consecutive SSE-alignments is started.

in the second sequence are very small, this could be just a coincidence. But it could also indicate some evolutionary event that took place in the past.

4 Conclusion and Outlook

Passta delivers a snapshot that may provide useful information: It shows how well a query sequence can be represented by PDB sequences, and at which positions. Since the classes of the chains that the aligned SSEs originate from are also displayed, some information on the secondary structure composition is available as well. Finally, the position information given for every aligned SSE w.r.t. its chain may indicate duplications, repeats or other evolutionary events.

Of course, some problems remain to be solved: Since Passta is presently based on pairwise sequence alignments, we can not expect it to find remote homologs in the 'twilight zone'. We also have to admit that some of the computed alignments are not very robust. Small variations of the jumpcost parameter can lead to large variations in the resulting alignment.

We plan to use a set of secondary structure specific substitution matrices as soon as possible. If the values for gap initiation and gap extension costs are wisely chosen, this should further improve the annotation quality of Passta.

Acknowledgments

We would like to thank Patrick May, Hans-Michael Kaltenbach, and Klaus-Bernd Schürmann for many interesting discussions; and the anonymous referee for some helpful remarks.

References

1. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215:403–410, 1990.
2. S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.*, 25:3389–3402, 1997.
3. C. Bannert. Systematic investigation of jumping alignments. *Technical Report*, 2003-05, 2003. `http://www.cebitec.uni-bielefeld.de/~bannert/pubs.html`.
4. H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The protein data bank. *Nucleic Acids Res.*, 28:235–242, 2000.
5. P. E. Bourne and H. Weissig. *Structural Bioinformatics*. Wiley Liss, 2003.
6. S. E. Brenner, P. Koehl, and M. Levitt. The astral compendium for protein structure and sequence analysis. *Nucleic Acids Res.*, 28:254–256, 2000.
7. J. M. Chandonia, N. S. Walker, L. Lo Conte, P. Koehl, M. Levitt, and S. E. Brenner. Astral compendium enhancements. *Nucleic Acids Res.*, 30:260–263, 2002.
8. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, 2nd Ed.* MIT Press / McGraw-Hill, 2001.
9. S. R. Eddy. Profile hidden Markov models. *Bioinformatics*, 14:755–763, 1998.
10. R. W. W. Hooft, C. Sander, and G. Vriend. The pdbfinder database: A summary of pdb, dssp and hssp information with added value. *CABIOS*, 12:525–529, 1996.
11. T. J. Hubbard, B. Ailey, S. E. Brenner, A. G. Murzin, and C. Chothia. SCOP: A Structural Classification of Proteins database. *Nucleic Acids Res.*, 27:254–256, 1999.
12. W. Kabsch and C. Sander. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22:2577–2637, 1983.
13. P. May, S. Barthel, and I. Koch. Ptgl - a web-based database application for protein topologies. *Bioinformatics*, 20:3277–3279, 2004.
14. A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: A structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, 247:536–540, 1995.
15. F. M. G. Pearl, D. Lee, J. E. Bray, I. Sillitoe, A. E. Todd, A. P. Harrison, J. M. Thornton, and C. A. Orengo. Assigning genomic sequences to cath. *Nucleic Acids Res.*, 28:277–282, 2000.
16. W. R. Pearson. Rapid and sensitive sequence comparison with FASTP and FASTA. In R. F. Doolittle, editor, *Molecular Evolution: Computer Analysis of Protein and Nucleic Acid Sequences*, volume 183 of *Meth. Enzymol.*, chapter 5, pages 63–98. Academic Press, San Diego, CA, 1990.
17. R. B. Russell, R. R. Copley, and G. J. Barton. Protein fold recognition by mapping predicted secondary structures. *J. Mol. Biol.*, 259:349–365, 1996.

18. J. Shi, T. L. Blundell, and K. Mizuguchi. FUGUE: Sequence-structure homology recognition using environment-specific substitution tables and structure-dependent gap penalties. *J. Mol. Biol.*, 310:243–257, 2001.
19. E. Shih and M.-J. Hwang. Alternative alignments from comparison of protein structures. *Proteins*, 56:519–527, 2004.
20. T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J. Mol. Biol.*, 147:195–197, 1981.
21. R. Spang, M. Rehmsmeier, and J. Stoye. A novel approach to remote homology detection: Jumping alignments. *J. Comp. Biol.*, 9:747–760, 2002.
22. M. S. Waterman and M. Eggert. A new algorithm for best subsequence alignments with application to trna-rrna comparisons. *J. Mol. Biol.*, 197:723–728, 1987.
23. Y. Ye, L. Jaroszewski, W. Li, and A. Godzik. A segment alignment approach to protein comparison. *Bioinformatics*, 19:742–749, 2003.