Soft Pattern Discovery in Pre-Classified Protein Families through Constraint Optimization

David Lesaint, Deepak Mehta, Barry O'Sullivan

Soft Pattern Discovery in Pre-Classified Protein Families through Constraint Optimization

David Lesaint[1], Deepak Mehta[2], and Barry O'Sullivan[2]

[1] LERIA, Université d'Angers, F-49045 Angers, France
david.lesaint@univ-angers.fr
[2] University College Cork, 4C, Cork, Ireland
d.mehta,b.osullivan@4c.ucc.ie

Abstract. A considerable effort has been invested in discovering patterns amongst protein sequences. This paper introduces the notion of *soft pattern* to characterize existing classes in a dataset. A soft pattern for a class is an exclusive set of subsequences, called c-blocks, which are common to the class and whose embeddings feature consistent mismatches. Soft patterns are less verbose than regular expressions and are not restricted to be linear as opposed to class signatures produced by multiple sequence alignment methods. We formalize a general soft pattern computation problem and present two variants enforcing either sequencing or non-replication of c-blocks in a pattern. We cast these problems into a lexicographic constraint optimization framework and present a two-stage procedure to solve the replication-free problem variant.

1 Introduction

Detecting patterns in amino acid sequences is critical to understand the relationship between the function and structure of proteins. This problem has been thoroughly investigated in Bioinformatics, notably via multiple sequence alignment (MSA) and pattern recognition approaches [1–6]. MSA methods insert gaps in input sequences (interpreted as amino acid indels) in order to align common blocks that have a limited amount of mismatch (interpreted as point mutations). By design, these methods only detect sequences of common blocks which is restrictive when analyzing proteins featuring different block orderings. Besides, MSA does not address the problem of enforcing pattern exclusivity when dealing with pre-classified sequences. The same applies to pattern recognition methods, and while the latter are less sensitive to block orderings, they may yield verbose or loose class signatures.

This paper proposes a constraint optimization approach to detect exclusive patterns in pre-classified protein families. The key requirements are addressed separately with constraints which gives the ability to use dedicated algorithms but also to integrate and reason about other dimensions of the problem (e.g., chemico-physical properties). The approach relies on the notion of *soft pattern* to accommodate point mutations and variable block orderings. A soft pattern for a class is an exclusive set of common subsequences where each subsequence features

the same point mutations across the class. We call *c-blocks* such subsequences and refer to point mutations as *hashes*. In order to facilitate biological interpretation and limit computational complexity, subsumption (*coverage*) between the c-blocks of a pattern is prohibited. Lexical subsumption and overlapping may also be prohibited to prevent *replication* or enforce *sequencing* of c-blocks.

On this basis, we formulate a core soft pattern discovery problem which we specialize into two different variants. The general problem is to detect exclusive and coverage-free patterns. The first variant detects replication-free patterns that are minimally exclusive and maximally refined while the second variant detects exclusive sequential patterns. All problems incorporate parametric constraints that bound the slack (i.e., the maximum number of consecutive hashes in c-blocks) and width of patterns (i.e., the minimum span of c-blocks) in order to discard degenerate solutions. Note that MSA problems may be cast as particular cases of the sequential pattern discovery variant.

The replication-free problem variant yields strong class characterisations since solution patterns are stripped of rendundant c-blocks while the remaining c-blocks cannot be specialized further, nor extended. We restrict our attention to this variant which we model as a lexicographic constraint optimization problem (COP). This COP addresses subsequence matching, commonality, hash-consistency, slack and width bounding, exclusivity, non-coverage and non equivalence of c-blocks through separate constraints. In particular, exclusivity is reducible to a set covering problem. The lexicographic objective function prioritizes minimum pattern cardinality over length maximality which ensures solutions are minimally exclusive and maximally refined soft patterns. We present a two-stage procedure to solve this COP. The procedure composes maximal c-blocks from minimal blocks before computing exclusive solution patterns.

The paper is organized as follows. Sec.2 discusses the notion of soft pattern and motivates the different pattern discovery tasks. Sec.3 formalizes the problems and introduces the constraint optimization model for computing replication-free patterns. Sec.4 presents the two-stage procedure for this problem. Sec.5 concludes. Due to lack of space, we refer the reader to [7] which includes proofs and presents a declarative implementation in Minizinc [8].

2 Soft Patterns

This section motivates the three pattern discovery problems and introduces the underlying concepts. Given a dataset composed of classes of protein sequences, the objective is to characterize a class by a *soft pattern*. A soft pattern is a set of *c-blocks* where a c-block corresponds to a subsequence that is common to the sequences of the class and whose adjacent characters are separated by the same number of mismatching characters in each sequence. We call *block* the embedding of a c-block in a sequence, and *hash* any mismatch inside a c-block. Note that a c-block cannot start nor end with a hash but no further constraint on hashes are assumed (allowed amino-acid substitutions and c-blocks starting or ending with hashes are not discussed here). Fig.1 shows a dataset made of

	Class A		*Class B*		
	protein	*protein*	*protein*	*protein*	*protein*
	ACADEEC	EECAEA	CADA	ADAAEEC	CAEEC
	1234567	123456	1234	1234567	12345
c-block	A#A	A#A	A#A	A#A	
c-block	CA	CA	CA		CA
c-block	EEC	EEC		EEC	EEC

Fig. 1. A soft pattern {A#A,CA,EEC} for class A.

two classes A and B. Class A includes proteins ACADECC and EECAEA and a soft pattern of three c-blocks is shown for it. The first c-block corresponds to the common subsequence AA with embeddings $\{1, 3\}$ in the first protein and $\{4, 6\}$ in the second, both sharing a hash in second position. This c-block is given signature A#A where # indicates a hash. The other two c-blocks are CA and EEC.

Beyond subsequence matching, commonality and hash-consistency, a key requirement is that patterns discriminate classes, that is, a pattern for a class should not match any "foreign" protein (i.e., any protein not in the class). *Pattern exclusivity* is defined differently based on the constraints one wishes to impose between the c-blocks of a pattern. We consider three constraints, namely, *non-coverage*, *non-replication* and *sequentiality*. Non-coverage means that no c-block embedding should contain another in a protein which is legitimate from biological and computational standpoints. In this case, a pattern is exclusive if any set of blocks in a foreign protein matching the c-blocks is itself coverage-free.

Optionally, a further restriction is to prohibit c-block *replication*. A c-block may indeed be repeated in a coverage-free pattern (i.e., c-blocks with the same signature) or replicated within larger c-blocks (e.g., a pattern containing AC#A and C#A). Replication subsumes coverage and determining exclusivity for replication-free patterns boils down to proving that every foreign protein is incompatible with one c-block. This is illustrated in Fig.1 where the pattern is replication-free and exclusive to class A since no protein of class B matches its three c-blocks.

Another alternative is to search for patterns whose c-blocks occur in the same order in each protein of the class. Since block embeddings may be abstracted as intervals, interval relations (e.g., Allen relations) may be used to characterize an ordering between blocks. The simplest and most intuitive relation is the precedence relation which guarantees that intervals do not overlap nor meet. Exclusivity holds in this case if any set of blocks in a foreign protein matching the c-blocks cannot be sequenced consistently with the pattern.

Different objectives may be pursued on top of these features. One such objective is to make patterns *minimally exclusive* and devoid of redundancy from a classification viewpoint. Minimal exclusivity is achieved by discarding c-blocks that do not contribute to making a pattern exclusive (notably, c-blocks that do not exclude any protein). This is the case for the pattern of Fig.1 since dropping any c-block yields a non-exclusive pattern. It would not be so if class B did not contain protein CAEEC, A#A being redundant in this case. Another objec-

tive is to try refining c-blocks as much as possible by substituting hashes with matching characters (*specialization*) or adding matching characters left or right, possibly introducing new hashes in doing so (*extension*). The pattern of Fig.1 is *maximally refined* in this sense.

Three indicators may also be used to assess pattern quality: *slack* (maximum number of consecutive hashes in c-blocks), *width* (minimum span of c-blocks) and *length* (total number of matching characters in the c-blocks). The width of a c-block is the sum of its slack and length while there is no such correlation for patterns as increasing the length of a pattern may increase the slack or decrease the width. Consistently with the need for maximality, preference goes to patterns with lower slack, greater width or length, all things being equal.

It is challenging though for domain experts to come up with a preference model aggregating all these features, criteria and objectives. On this basis, we formulate a general *soft pattern discovery problem* (SPD) and two variants. The SPD consists in computing exclusive and coverage-free patterns for a class. Note that SPD solutions may feature replicated c-blocks. The SPD enforces two parametric constraints, namely, an upper bound on slack to discard loose patterns and a lower bound on width to discard short patterns. The first variant, called *replication-free soft pattern discovery problem*, consists in computing SPD solution patterns that are minimally exclusive, maximally refined and non equivalent for replication. Such solutions are replication-free and minimal exclusivity boils down to a minimal set covering problem. The second variant, called *sequential soft pattern discovery problem*, consists in searching for sequential SPD solution patterns. Minimal exclusivity and c-block maximality may conflict under a precedence ordering which is why neither feature is imposed in this variant.

3 Soft Pattern Discovery Problems

We use the following notations. For $n \in \mathbb{N}$, $[n]$ denotes the *range* $\{i \in \mathbb{N} \mid 1 \leqslant i \leqslant n\}$, $|t|$ denotes the number of elements of a tuple or a set t and $[t]$ denotes the range $[|t|]$, t_i denotes the i-th element of a tuple t for all $i \in [t]$, and $f(A)$ denotes the image of a function $f : A \to B$ (i.e., $f(A) = \{f(i) \mid i \in A\}$). Σ denotes a finite *alphabet* and Σ^* the set of finite strings that are constructed from the characters of Σ by concatenation. Tuple notations will be used for strings.

A *class over* Σ is a tuple of strings belonging to Σ^* and a *dataset over* Σ is a tuple of classes over Σ. A string x has *length* n or is *n-long* if it consists of n, not necessarily distinct, characters from Σ. A string y is a *substring* of a string x if there exist not necessarily distinct and possibly empty strings $v_1, v_2 \in \Sigma^*$ such that $v_1 y v_2 = x$. A k-long string $y = y_1 \dots y_k$ is a *subsequence* of a string x if there exist $k+1$ not necessarily distinct and possibly empty strings $v_1, \dots v_{k+1} \in \Sigma^*$ such that $v_1 y_1 \dots v_k y_k v_{k+1} = x$. We denote this fact by $y \preceq x$. Let $y \preceq x$, an embedding of y in x is a strictly increasing function $\mu : [y] \to [x]$ such that $y_i = x_{\mu(i)}$ for all $i \in [y]$. Note that a subsequence may have multiple embeddings. A string y is a *common subsequence* of a class x if $y \preceq x_i$ for all $i \in [x]$.

A *block* for a string x is a triple $\langle \mu, y, x \rangle$ such that $y \preceq x$ and μ is an embedding of y in x. We associate to a block $b = \langle \mu, y, x \rangle$ a *hash function* $\gamma(b) : [y] \to [|x| - |y|]$ defined by $\gamma(b)(|y|) = 0$ and $\gamma(b)(i) = \mu(i+1) - \mu(i) - 1$ $(1 \leqslant i < |y|)$. For instance, `CDA` is a subsequence of `CACDEAC` that determines two blocks, the left-most one `C##D#A` verifying $\mu([y]) = \{1, 4, 6\}$ and $\gamma([y]) = \{2, 1, 0\}$. Let $b = \langle \mu, y, x \rangle$ and $b' = \langle \mu', y', x' \rangle$ be two blocks, we say that b and b' are *compatible* if $y = y'$ and $\gamma(b) = \gamma(b')$. In other words, compatible blocks embed a common subsequence with identical hashes. Let $b = \langle \mu, y, x \rangle$ and $b' = \langle \mu', y', x \rangle$ be two blocks for the same string x, we say that b *covers* b' if $\mu([y]) \supseteq \mu'([y'])$ and that b *replicates* b' if there exists a block b'' for x, not necessarily distinct from b', compatible with b' and covered by b. Coverage is a particular case of replication. For instance, block `ACD` covers and replicates the right-most block `CD` in string `CDACD` but only replicates the left-most block `CD`.

A *c-block* for a class x is a $|x|$-tuple t of compatible blocks such that $t_i = \langle \mu_i, y_i, x_i \rangle$ for all $i \in [x]$. That is, a c-block determines a common subsequence in a class whose embeddings have identical hashes. We say that a c-block is *compatible* with a block b if its blocks are compatible with b; and that it is *incompatible* with a string z if there is no block for z compatible with it. Let t and t' be two c-blocks for a class x, we say that t *covers* t' (respectively, t *replicates* t'), denoted $t \geqslant t'$ (resp., $t \succeq t'$), if there exists $i \in [x]$ such that t_i covers t'_i (resp., t_i replicates t'_i). Replication is a partial order that subsumes coverage which is itself non-transitive. Both relations preserve incompatibility, i.e., if $t \succeq t'$ and t' is incompatible with string z then t is incompatible with z. We introduce the following relations: $t < t' \Leftrightarrow (t \leqslant t' \wedge \neg(t \geqslant t'))$, $t \cong t' \Leftrightarrow (t \preceq t' \wedge t \succeq t')$, and $t \prec t' \Leftrightarrow (t \preceq t' \wedge \neg(t \succeq t'))$. $\prec$ is a strict partial order as opposed to $<$ and $\cong$ is an equivalence relation (equivalent c-blocks have signatures over $\Sigma \cup \{\#\}$).

A *pattern* for a class x is a set of c-blocks for x. We say that a pattern is compatible with a string z if one can replicate its c-blocks in z without coverage. Formally, let c be a dataset and p be a pattern for class c_i for some $i \in [c]$, p is *compatible* with a string z if there exists a set q of blocks for z such that (1) for all $t, t' \in p$, $t \neq t'$, there exists $b \in q$ and $b' \in q$ such that $b \neq b'$, t is compatible with b and t' is compatible with b', and (2) for all $b, b' \in q$, $b \neq b'$, b does not cover b'. If there is no replication in p, it suffices to show that each c-block of p is compatible with z. We say that p is *exclusive* to c_i wrt. c if for all $j \in [c]$ such that $j \neq i$, for all $k \in [c_j]$, p is incompatible with c_{j_k}. We say that p is *minimally exclusive* for a class c_i wrt. a dataset c if any pattern strictly included in p is not exclusive for c_i wrt. c.

We denote by λ, σ and ω the length, slack and width functions used for blocks, c-blocks or patterns. Let $b = \langle \mu, y, x \rangle$ be a block, $\lambda(b) = |y|$, $\sigma(b) = \max_{i \in [y]} \gamma(b)(i)$ and $\omega(b) = \mu(|y|) - \mu(1) + 1$. Let t be a c-block, $\lambda(t) = \lambda(t_i)$, $\sigma(t) = \sigma(t_i)$ and $\omega(t) = \omega(t_i)$ where $i \in [t]$. Let p be a pattern, $\lambda(p) = \sum_{t \in p} \lambda(t)$, $\sigma(p) = max_{t \in p} \sigma(t)$ and $\omega(p) = \min_{t \in p} \omega(t)$. Let $k \in \mathbb{N}$ and x denote a block, c-block or pattern, we say that x is *k-loose* if $\sigma(x) \leqslant k$; x is *strict* if it is 0-loose; and x is *k-wide* if $\omega(x) \geqslant k$. Let $l \in \mathbb{N}$ and $w \in \mathbb{N}$, a c-block t for a class x such that $\sigma(t) \leqslant l$ and $\omega(t) \geqslant w$ is $<^l_w$*-minimal over* x (resp. $<^l_w$*-maximal*,

$\prec_w^l$*-minimal*, $\prec_w^l$*-maximal*) if there is no c-block t' for x such that $\sigma(t') \leqslant l$, $\omega(t') \geqslant w$ and $t > t'$ (resp., $t < t'$, $t \succ t'$, $t \prec t'$).

The soft pattern discovery problem (SPD) consists in determining slack- and width-bounded patterns that are exclusive and coverage-free. The replication-free SPD (RSPD) requires that c-blocks be maximal for $\prec_w^s$ and non equivalent to prevent replications (non-equivalence may be dropped to allow repetitions). The sequential SPD (SSPD) requires that c-blocks be consistently and totally ordered over the class based on the following relation over embeddings: block $b = \langle \mu, y, x \rangle$ *precedes* block $b' = \langle \mu', y', x \rangle$ if $\mu(|y|) + 1 < \mu'(1)$. Let t and t' be two c-blocks over a class x, we say that t *precedes* t' if for all $i \in [x]$, t_i precedes t'_i, and that a pattern p is *sequential* if *precedes* is a total ordering over p.

Definition 1 (SPD, RSPD, SSPD). *Let C be a dataset over an alphabet A, $i \in [C]$, $s \in \mathbb{N}$, and $w \in \mathbb{N}^*$. A solution to a SPD $\langle A, C, i, s, w \rangle$ is an exclusive, coverage-free, s-loose and w-wide pattern for C_i. A solution to a RSPD $\langle A, C, i, s, w \rangle$ is a minimally exclusive pattern of $\prec_w^s$-maximal and non-equivalent c-blocks for C_i. A solution to a SSPD $\langle A, C, i, s, w \rangle$ is a sequential, exclusive, s-loose and w-wide pattern for C_i.*

We propose a lexicographic constraint optimisation model for the RSPD that computes solution patterns of minimum cardinality and, amongst those, of maximum length. The model substitutes maximality and minimality constraints with the requirement that c-blocks be maximal according to the lexicographic ordering prioritizing minimum cardinality over maximum length.

Lemma 1 (RSPD as a COP). *Let $P = \langle A, C, i, s, w \rangle$ be a RSPD, Π the set of patterns for C_i, and $p \in \Pi$. p is a minimum cardinality solution to P if and only if it satisfies the following conditions:*

1. *slack and width bounds: $\sigma(p) \leqslant s$ and $\omega(p) \geqslant w$;*
2. *exclusivity: p is exclusive to C_i wrt. C;*
3. *non-equivalence: for all $t \in p$, $u \in p$ s.t. $t \neq u$, $t \not\cong u$;*
4. *non-coverage: for all $t \in p$, $u \in p$ s.t. $t \neq u$, $\neg(t < u)$;*
5. *maximality for $\prec_{lex}$: for all $q \in \Pi$ s.t. q satisfies (1-4), $\neg(p \prec_{lex} q)$ where $p \prec_{lex} q$ iff $(|q| \geqslant |p| \;\Rightarrow\; (|q| = |p| \;\wedge \lambda(q) > \lambda(p)))$ holds true.*

The above result generalizes to the case where we include slack minimality as the least preferred criterion in the lexicographic objective function. Since RSPDs prohibit replication, the maximum number of solutions to a RSPD is (loosely) bounded by the maximum number of blocks in the smallest protein of the class C_i. The following result formulates the bound in the case of 1-loose patterns.

Lemma 2. *Let $\phi = \frac{1+\sqrt{5}}{2}$, $\psi = \frac{1-\sqrt{5}}{2}$, $n \in \mathbb{N}$ and $\beta(n)$ be the number of 1-loose blocks for a n-long string. $\beta(n) = \frac{\phi^{n+4} - \psi^{n+4}}{\sqrt{5}} + n - 3$ and $\lim_{n\to\infty} \beta(n) = \frac{\phi^{n+4}}{\sqrt{5}}$.*

More generally, the number of blocks of slack k that span a string of length n is the Fibonacci sequence of order k (where each element is the sum of the

previous k elements). The closed-form for the n-th element of the sequence is $[\frac{r^{n-1}(r-1)}{(k+1)r-2k}]$ where [] denotes the nearest integer function and r is the limit of the ratio between successive terms as n increases. r corresponds to the root of equation $x + x^{-k} = 2$ near to 2 and it approaches 2 as k increases.

4 A Two-Step Approach to Solving (R)SPD

As (R)SPD involves finding a pattern consisting of one or more maximal c-blocks for a class C_k, we propose a two-step approach where first we compute all the maximal c-blocks for C_k and then compute an exclusive pattern of minimum cardinality. We describe these two steps briefly.

Computing maximal c-blocks. To compute the set of maximal c-blocks, we first compute all blocks of length 2 that are common to the proteins of class C_k. This is done by first selecting the protein having the minimum size and then verifying for each valid block of length 2 whether it is common to all the proteins or not. Once done, we know all the minimal length blocks and their starting positions in the smallest protein of the class. We then compute all maximal blocks of the protein by composing the minimal length blocks while ensuring that each of them is involved in at least one c-block. Finally, we compute the set of all maximal c-blocks based on the maximal blocks and ensure that none of them covers another.

Computing optimal patterns. Once we have all the maximal c-blocks, we formulate a constraint optimization problem for computing an optimal pattern, i.e., an exclusive pattern of minimum cardinality. Let A denote the set of the maximal c-blocks for class C_k and F denote the set of foreign proteins (i.e., proteins not in C_k). For each protein $i \in F$, we compute the subset of A containing the maximal c-blocks whose signature is not matched by protein i. This set is denoted by $E_i \subseteq A$. For each combination of a protein $i \in F$ and a maximal c-block $j \in E_i$ a Boolean variable x_{ij} is created which denotes whether j is used to exclude i. Another Boolean variable y_j is created that denotes whether the maximal c-block j is part of the pattern or not. For each protein $i \in F$, we want to select at least one maximal c-block whose signature is not matched by i, i.e., $\sum_{j \in E_i} x_{ij} \geqslant 1$. A maximal c-block is selected if it is used to exclude a protein i, i.e., $y_j \geqslant max_{i \in F} x_{ij}$. The objective is to minimize the number of maximal c-blocks that are selected for class C_k, i.e., $\min \sum_{j \in A} y_j$.

Notice that the formulation is equivalent to that of a minimum set covering problem. This model only works for RSPD as it does not use the number of times a c-block signature is matched by a foreign protein to enforce exclusivity. We remark that it is possible to have a multiple c-blocks with same signature. Indeed, it is possible that a c-block signature is matched by a foreign protein but not as many times as there are maximal c-blocks in A sharing this signature. In this case, the model will consider each of these c-blocks as compatible whereas they are incompatible as a whole.

Let s_n be the number of maximal c-blocks associated with a signature s, and let $\{c_{s_1}, \ldots, c_{s_n}\} \subseteq A$ be some permutation of those maximal c-blocks. For solving SPD, we additionally associate each maximal c-block c_{s_j} with a number j, and the constraint that if c_{s_j} is selected then at least j number of c-blocks associated with the same signature must be selected. These constraints are enforced through a set of dependencies.

Our preliminary results using SPD are shown in Table 1 which suggest that the presented approach is indeed scalable for handling large size instances. We investigated with two databases: Late Embryogenesis Abundant Proteins (LEAP) and Small Heat Shock Proteins (sHSP). The number of classes and the total number of proteins in these classes is mentioned in the columns labelled as `nclasses` and `nproteins` respectively. For LEAP 6 out of 12 classes and for sHSP 3 out of 23 classes were unsatisfiable as they do not have any exclusive SPD patterns. The maximum (and the minimum) number of maximal c-blocks, slack and length of an optimal exclusive patterns associated with the classes of each database is shown in the columns labelled as `cardinality`, `slack` and `length` respectively.

Table 1. Results of LEAP and sHSP instances obtained using SPD

name	nclasses	nproteins	#unsat	cardinality	slack	length
LEAP	12	1066	6	5 (1)	4 (0)	13 (4)
sHSP	23	2244	3	10 (1)	25 (0)	20 (4)

5 Conclusion

We have introduced the notion of soft pattern to characterize and discriminate classes of protein sequences. Three pattern discovery problems have been formalized to prevent c-block coverage, replication or overlapping. We have shown that minimal exclusion and maximal refinement are compatible objectives when computing replication-free patterns. The principles of a lexicographic constraint optimization model have been laid out and a two-stage procedure has been sketched. Future work involves carrying out experiments on two existing datasets of unstructured and highly structured proteins [9–12].

References

1. Gusfield, D.: Algorithms on Strings, Trees and Sequences. Computer Science and Computational Biology. Cambridge University Press (2008)
2. Seiler, M. et al.: The 3of5 web application for complex and comprehensive pattern matching in protein sequences. BMC Bioinformatics, pp. 7–144 (2006)

3. Bailey, T.L. et al.: MEME SUITE: tools for motif discovery and searching. Nucleic Acids Research, 37:W202W208 (2006)
4. Uversky, V.N., Dunker, A.K.: Understanding protein nonfolding. Biochim. Biophysi. Acta 1804, pp. 1231–1264 (2010)
5. Grant, C.E., Bailey, T.L., Noble, W.S.: FIMO: Scanning for occurrences of a given motif. Bioinformatics, vol. 27, pp. 1017–1018 (2011)
6. Dinkel et al.: ELM the database of eukaryotic linear motifs. Nucleic Acids Res., vol. 40: D242-D251 (2012)
7. Lesaint, D., Mehta, D., O'Sullivan: Soft Pattern Discovery in Pre-Classified Protein Families through Constraint Optimization. Technical report (2013)
8. Nethercote, N. et al.: MiniZinc: towards a standard CP modelling language, Princ. and Pract. of Constraint Programming (CP'07), pp. 529–543, Springer-Verlag (2007)
9. Hunault, G., Jaspard, E.: The Late Embryogenesis Abundant Proteins DataBase. `http://forge.info.univ-angers.fr/~gh/Leadb/index.php` (2013)
10. Hunault, G., Jaspard, E.: LEAPdb: a database for the late embryogenesis abundant proteins. pp. 11–221, BMC Genomics (2010)
11. Jaspard, E., Macherel, D., Hunault, G.: Computational and statistical analyses of amino acid usage and physico-chemical properties of the twelve late embryogenesis abundant protein classes. PLoS ONE 7:e36968 (2012)
12. Hunault, G., Jaspard, E.: The Small Heat Shock Proteins Database. sHSPdb. `http://forge.info.univ-angers.fr/~gh/Shspdb/index.php` (2013)