Combining constraint processing and pattern matching to describe and locate structured motifs in genomic sequences

Patricia Thébault, Simon de Givry, Thomas Schiex, and Christine Gaspin
{pat,degivry,tschiex,gaspin}@toulouse.inra.fr
INRA Biometrics and Artificial Intelligence
Toulouse, France

Abstract

In molecular biology and bioinformatics, searching RNA gene occurrences in genomic sequences is a task whose importance has been renewed by the recent discovery of numerous functional RNA, often interacting with other ligands. Even if several programs exist for RNA motif search, no program exists that can represent and solve the problem of searching for occurrences of RNA motifs **in interaction with other molecules**.

In this paper, we present a CSP formulation of this problem. We represent such RNA as structured motifs that occur on more than one sequence and which are related together by possible hybridization. Together with pattern matching algorithms, constraint satisfaction techniques have been implemented in a prototype MilPat and applied to search for tRNA and snoRNA genes on genomic sequences. Results show that these combined techniques allow to efficiently search for interacting motifs in large genomic sequences and offer a simple and extensible framework to solve such problems.

1 Introduction

Our understanding of the role of RNA has changed in recent years. Firstly considered as being simply the messenger that converts genetic information from DNA into proteins, RNA is now seen as a key regulatory factor in many of the cell's crucial functions, affecting a large variety of processes including plasmid replication, phage development, bacterial virulence, chromosome structure, DNA transcription, RNA processing and modification, development control and others (for review [Storz, 2002]). Consequently, the systematic search of non-coding RNA (ncRNA) genes, which produce functional RNAs instead of proteins, represents an important challenge.

RNA sequences can be considered as texts over the four letter alphabet $\{A,C,G,U\}$. Unlike double-stranded DNA, RNA molecules are almost exclusively found in an oriented (left or 5' to right or 3') single-stranded form and often fold into more complex structures than DNA by making use of so called complementary internal sequences. This characteristic allows different regions of the same RNA strand (or of several RNA strands) to fold together via a variety of interactions to build structures that are essential for the biological function. The level of organization relevant for biological function corresponds to the spatial organization of the entire nucleotides chain and is called the tertiary structure. The most prevalent interactions which stabilize folded molecules are stacking and hydrogen bonding between nucleotides on strands oriented in antiparallel directions. Similarly to what exists in DNA, hydrogen bonds appear mostly between specific pairs of nucleic acids to form *G–C* and *C–G* or *A–U* and *U–A* bonds. Therefore the interactions inside an RNA molecule usually involve one part of a molecule and the nucleic acid complement of a another part of the same molecule (for example, 5'-ACUCGA-3' and 5'-UCGAGU-3'), and the two antiparallel regions bind together.

All together, these interactions define the molecule three-dimensional structure which is essential to characterize its function and interactions with other molecules. Due to the difficulty of determining such three dimensional RNA structures, one first explores the so-called RNA secondary structure, a simplified model of the RNA three dimensional tertiary structure.

This secondary structure gives only a subset of those interactions represented by *C–G*, *G–C*, *A–U*, and *U–A* pairs and provides an important constraint for determining the three dimensional structure of RNA molecules.

An RNA molecule secondary structure can be represented on a circular planar graph where the N nucleotides of the sequence are represented as vertices and are connected by edges representing either (along the circle) covalent bonds between successive nucleotides in the RNA sequence or (inside the circle) hydrogen bonds between nucleotides from different regions. Such a graph gives rise to characteristic secondary structural elements (see Fig. 1) such as helices (a succession of paired nucleotides), and various kinds of loops (unpaired nucleotides surrounded by helices).

A more complete definition of secondary structure of RNA allows for crossing edges in the representation graph making possible the representation of another type of helix usually called a pseudoknot (see Fig. 1). RNA structures can also include nucleotide triples inside triple helices...

This definition extends the usual definition which is often limited to planar structures (therefore excluding pseudo-

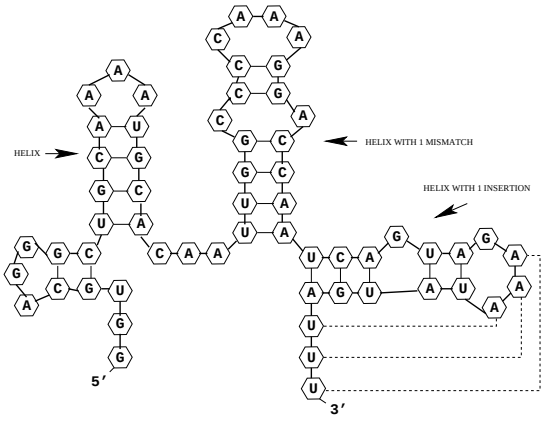

Figure 1: A representation of a secondary structure as a planar graph. Thick edges represent covalent bonds. Thin edges represent hydrogen interactions. Dotted edges represent a pseudoknot. Helices may contain local mismatches which cover three different types of errors which are: insertions of nucleotide(s), also called bulges when only on one side, deletions when the insertion is on the opposite side of the helix and internal loops, when nucleotides are located on both sides of the same helix. Insertion and deletion of nucleotides are considered as symmetric operations, an insertion on one side corresponding to a deletion on the other complementary region.

knots and multiple helices) and is always restricted to intra-sequences interactions.

In this paper, we use the extended definition where the secondary structure of an RNA gene is defined as the set of paired nucleotides which appear in the folded RNA, including possible pseudoknots, triple helices but also duplexes which are possible bindings forming helices with other RNA molecules.

Screening a sequence database with tools designed for sequence similarity search quickly reveals similarities between the query sequence and a range of database sequences. This can be achieved for ribosomal rRNA sequences and other ncRNAs recently reported in the literature (although it is difficult to establish the beginning and end of the RNA in question). But the nucleotide sequence of the RNA itself is poorly conserved, the observation that the functionally important structural regions are usually conserved in an RNA family (see for example Fig. 2) allows one to search for those elements that characterize the family more precisely.

Thus, the information contained both in the sequence itself and the secondary (tertiary) structure can be viewed as a biological signal to exploit and search for. Thus, whatever the method, it appears necessary to include both conserved primary sequence elements and higher order structure elements as signals to screen for. These common structural characteristics can be captured by a signature that represents the structural elements which are conserved inside a set of related RNA molecules.

We focus here on the problem of searching for new members of a gene family given their common signature. Solving this problem requires (1) to be able to formalize what a signature is and what it means for such a signature to occur in a sequence (2) to design algorithms and data-structures that can efficiently look for such occurrences in large sequences. For sufficiently general signatures, this is an NP-complete problem [Vialette, 2004] that combines combinatorial optimization and pattern matching issues.

Traditionally, two types of approaches have been used for RNA gene finding: signatures can be modelled as stochastic context free grammars (excluding pseudo-knots or complex structures) and then searched using relatively time consuming dynamic programming based parsers. This is e.g. used in [Sakakibara *et al.*, 1994; Eddy and Durbin, 1994] for RNA genes or in [Bockhorst and Craven, 2001] for terminators.

Another approach defines a signature as a set of interrelated motifs. Occurrences of the signature are sought using simple pattern-matching techniques and exhaustive tree search. Such programs include RnaMot [Gautheret *et al.*, 1990], RnaBob [Eddy, 1996], PatScan [Dsouza *et al.*, 1997], Palingol [Billoud *et al.*, 1996] and RnaMotif [Macke *et al.*, 2001]. Although most allow pseudo-knots to be represented, they have very variable efficiencies and are all restricted to single RNA molecule signatures.

In this paper, we clearly separate the combinatorial aspects from the pattern matching aspect by modelling a signature as a CSP. The CSP model captures the combinatorial features of the problem while the constraints use pattern matching techniques to enhance efficiency. This combination offers an elegant and simple way to describe several RNA motifs in interaction and a general purpose efficient algorithm to search for occurrences of such motifs.

2 Methods

The CSP formalism (see e.g. [Dechter, 2003]) is a powerful and extensively used framework for describing combinatorial search problems in artificial intelligence and operations research. This is usually well adapted to the definition of mathematical problems raised by molecular biology (see [Gaspin and Westhof, 1994; Muller *et al.*, 1993; Altman *et al.*, 1994; Major *et al.*, 1991; Barahona and Krippahl, 1999]) and has been used to model the structured motif search problem in [Eidhammer *et al.*, 2001; Policriti *et al.*, 2004].

2.1 Structured motifs as CSPs

The elements that may characterize an RNA gene family are usually described:

- in terms of the gene sequence itself (e.g. it must contain some possibly degenerated pattern);
- in terms of the structures the sequence creates: loops, helices, hairpins and possible duplexes with other molecules;
- by specifying how these various elements are positioned relatively to each other.

A possible occurrence of such a structured motif on a genomic sequence can be described by the positions of the various elements on the genomic sequence. A true occurrence is

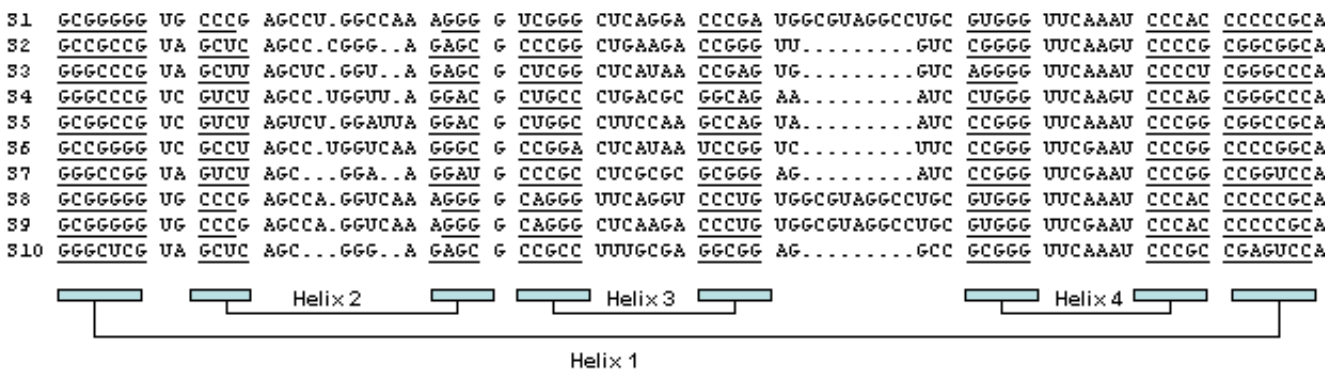

Figure 2: Alignment of a subset of ten sequences of the tRNA family extracted from the RFAM RNA database (http://www.sanger.ac.uk/cgi-bin/Rfam). Each line gives the tRNA gene sequence. Both sides of each helix are underlined for each sequence of the alignment. Consensus helices are identified by boxes at the end of the alignment. tRNA genes include four helices corresponding respectively to helix 1 called A-stem (7 nucleotide pairs), helix 2 called D-stem (from 3 to 4 nucleotide pairs), helix 3 called C-stem (5 nucleotide pairs) and the last fourth helix, called, T-stem (5 nucleotide pairs). Six loops corresponding respectively to the single strand between A-stem and D-stem (sequence UN with U invariant), D-loop (4 to 14 nucleotides), the single strand between D-stem and C-stem (one nucleotide), C-loop (6 to 60 nucleotides), the single strand between C-stem and T-stem (also called V-loop, 2 to 22 nucleotides), T-loop (NUC) allow to build a signature of the family. Note several hundred tRNA sequences are now available from biological databanks (see in particular RFAM).

such that the required patterns, structures actually appear in the genomic sequence and are correctly positioned relatively to each other. Note that a genomic sequence is represented as a string defined over the RNA alphabet $\{A,U,G,C\}$.

A natural CSP model emerges from this description: the variables will represent the positions on the nucleotide sequence of the elements of the description. More formally, each variable $x_i \in X$ will represent a position on an associated RNA sequence (denoted t_i). The initial domain of variable x_i, unless otherwise stated, will therefore be equal to $[1,|t_i|]$. In order to represent information on required patterns, structures and on relative positions of these elements, constraints will be used. To describe a constraint we separate the *variables* $x_i,...,x_j$ involved in the constraint (its scope) and possible extra *parameters* $p_1,...,p_k$ that influence the actual combination of values that are authorized by the constraint. Such a constraint will be denoted as $\mathsf{name}[p_1,...,p_k](x_i,...,x_j)$. We now introduce the basic constraint types which are useful for RNA signature expression:

`composition`$[\mathsf{word},\mathsf{error},\mathsf{type_{err}}](x_i)$

this unary constraint is satisfied iff some given sequence (a pattern) occurs at position x_i on sequence t_i. The pattern that must occur is specified by the following constraint parameters:

- word is a word on the so-called IUPAC alphabet which includes meta-characters that match several characters of the RNA alphabet.
- error specifies the maximum number of tolerated mismatches between an occurrence and the specified string.
- $\mathsf{type_{err}}$ indicates if the error count is interpreted under the Hamming or Levenstein distance metric [Smith and Waterman, 1981].

An example of possible use of this constraint is illustrated in Fig. 3(1) where variable x_1 is constrained to a position where the AGGGCUAG pattern must appear with no error. A position satisfying this constraint (or occurrence of the pattern) is indicated by the arrow.

`distance`$[\mathsf{l_{min}},\mathsf{l_{max}}](x_{i_1},x_{i_2})$

this binary constraint is use to enforce the relative position of elements. It is satisfied iff

$$\mathsf{l_{min}} \leq x_{i_2} - x_{i_1} \leq \mathsf{l_{max}}$$

The parameters $\mathsf{l_{min}},\mathsf{l_{max}}$ specify the bounds for the difference between the two position variables. It is a simple usual arithmetic constraint.

`helix`$[\mathsf{rule},\mathsf{error},\mathsf{type_{err}},\mathsf{l_{min}},\mathsf{l_{max}},\mathsf{b_{min}},\mathsf{b_{max}}](x_{i_1},x_{i_2},x_{i_3},x_{i_4})$

this 4-ary constraint is used to enforce the existence of an helix between the sequence regions delimited by $[x_{i_1},x_{i_2}]$ and $[x_{i_3},x_{i_4}]$. This constraint assumes that the four variables are related to the same sequence (it models intra-sequence interactions) and each region represents a substring of this sequence. The length and distances between these regions are also constrained. The constraint must be specified by the following parameters:

- rule: a binary relation on the RNA alphabet that characterizes which pairs of nucleotides are allowed inside an helix. For an RNA helix, one typically uses Watson-Crick (A-U and G-C) pairs,possibly extended with Wobble (G-U) pairing.
- error: the maximum number of tolerated mismatches between the two regions (nucleotides that do not satisfy the previous paring relation).
- $\mathsf{type_{err}}$: the Hamming or Levenstein distance metric for error counts.

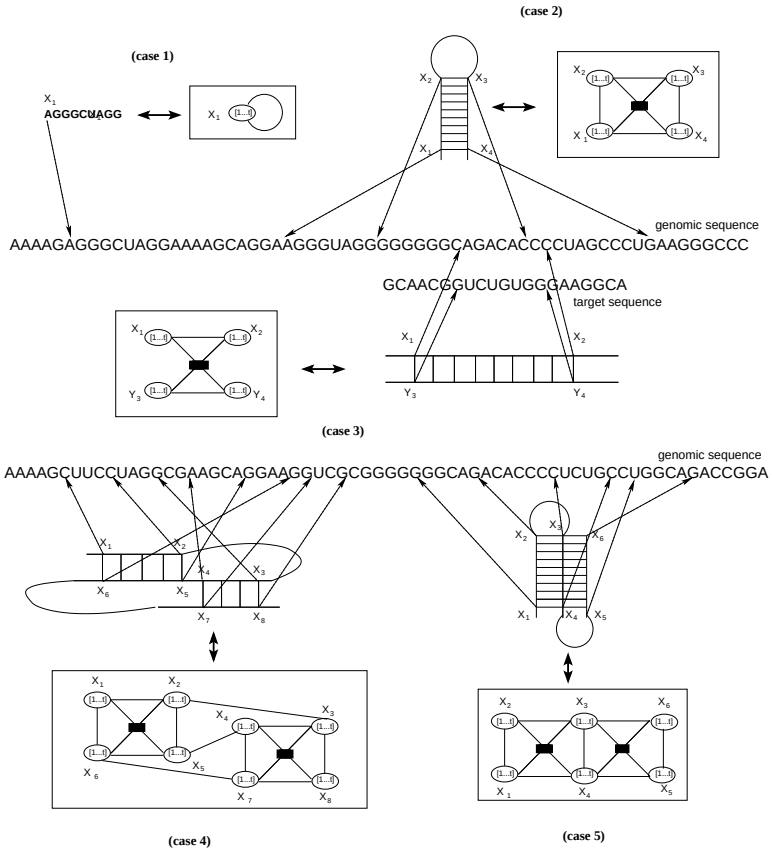

Figure 3: Basic constraints. (case 1): occurrence of a pattern at one position (variable). The constraint graph contains one variable with a unary constraint represented by a loop. (case 2): an helix and a loop defined by two related segments separated by specified lengths. The constraint graph contains four variables, four implicit distance constraints represented by edges and one hyper-edge (for the helix constraint) connecting all four variables with a rectangle in the middle. (case 3): a duplex composed of two independent substrings (from two sequences). The constraint graph is similar to the previous one (two distance constraints are removed). (cases 4 and 5): two helix constraints can describe a pseudo-knot (4) or a triple helix (5).

- $\mathsf{l}_{\mathsf{min}}, \mathsf{l}_{\mathsf{max}}$: the interval specifying possible lengths of the two substrings.

- $\mathsf{b}_{\mathsf{min}}, \mathsf{b}_{\mathsf{max}}$: the interval specifying the possible distance between the two substrings (*i.e.*, $x_{i_3} - x_{i_3}$).

This constraint is illustrated in Fig. 3(2), involving variables x_1, x_2, x_3 and x_4. Assuming Watson-Crick pairing, no error and suitable lengths, the constraint is satisfied for the values indicated by arrows on the sequence below.

`duplex`$[\mathsf{l}_{\mathsf{min}}, \mathsf{l}_{\mathsf{max}}](x_1, x_2, y_3, y_4)$

this 4-ary constraint is used to enforce the existence of a (Watson-Crick based) duplex between the regions delimited by $[x_1, x_2]$ and $[y_3, y_4]$. Although semantically equivalent to the previous one, it does not assume that the two substrings represented by the two regions belong to the same sequence. This has important computational impact. Only Watson-Crick pairing is considered. This constraint is used to model RNA-RNA interactions between possibly different molecules.

- $\mathsf{l}_{\mathsf{min}}, \mathsf{l}_{\mathsf{max}}$: the interval specifying possible lengths of the two substrings.

The constraint is illustrated in Fig. 3(3) involving x_1, x_2 (on one sequence) and y_3 and y_4 on another sequence. Values satisfying the constraint (an occurrence) is indicated by the arrows.

Note that together these constraints can describe more complex structures like pseudo-knots (Fig. 3(4)), triple helices (Fig. 3(5)), and so on.

The flexibility of the CSP formalism using simply the four previous basic constraints can be illustrated on famous RNA gene families. The **tRNA** signature is represented in Fig. 4 where tRNA genes include four helices. The corresponding CSP is build from 16 variables (the variable numbering follows the 3' $\rightarrow$ 5' orientation) with 15 `distance` constraints (one constraint between each successive pair of variables), 2 `composition` constraints and 4 `helix` constraints.

The same process can be applied to the **snoRNA** signature depicted in Fig. 5. snoRNA genes include a C box (RUGAUGA) with one error allowed, a single strand from 22 to 44 nucleotides, a duplex with a target RNA from 9

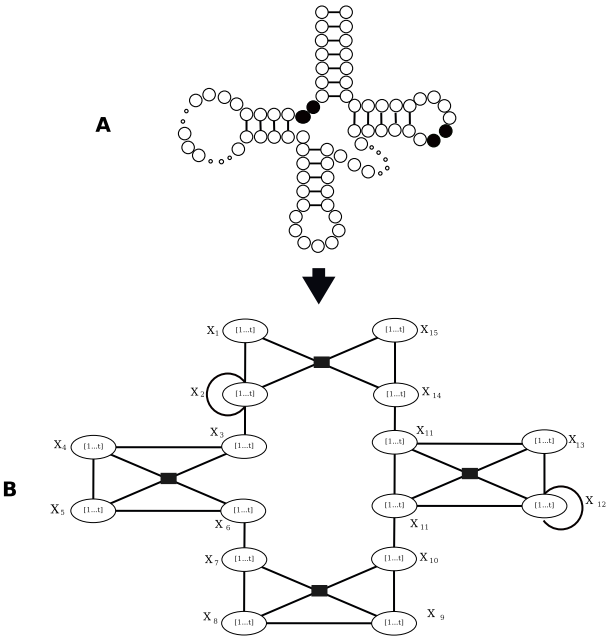

Figure 4: (A) Signature of tRNA genes family. White circles : nucleotides with unknown composition, black circles : known composition, little circles : number of nucleotides given by an interval, and edge : interaction between two nucleotides. (B) Corresponding CSP model.

to 15 nucleotides and a D box (CUGA) with one error allowed. The corresponding CSP is build from 4 variables corresponding to positions on the genomic sequence and a pair of additional variables associated with the target RNA. The first set of variables is linked with 3 `distance` constraints and 2 `composition` constraints. The second set with one `distance` constraint. Both sets are connected through one `duplex` constraint.

2.2 Algorithms and implementation

Given such CSPs, our problem is to find **all solutions**. Compared to usual applications of the CSP formalism, this one is characterized by the potential huge domain size (the length of a complete pseudo-molecule can be greater than several million of nucleotides) and its specific constraint types (except for the `distance` constraint which is a usual arithmetic constraint). For efficiency and memory space reasons, it is not possible to represent variable domains exhaustively and to enforce arc consistency on them. As it is done in Constraint Programming [Dechter, 2003], we represent the domain of each variable x_i by an interval $[lb_i, ub_i]$ and reason only on domain bounds as done in arc-bound consistency [Lhomme, 1993]. This limited bound filtering is done at each node inside a usual tree search algorithm. For n-ary constraints, the typical form of local consistency used enforces the fact that the bounds in the domain of one variable in the constraint scope must participate in at least one tuple that is authorized by the constraint and the other domains. The exploration method we used is a depth-first search algorithm with a refutation mechanism (during backtracks, it propagates the removal of values

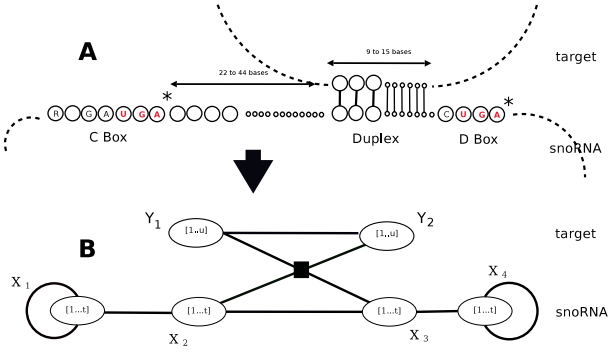

Figure 5: Signature of snoRNA genes family including its target interaction.

already explored).

Dedicated constraint propagation

For each type of constraint, we developed specific filtering algorithm using appropriate pattern matching algorithms (except for the `distance` constraint where we used the filtering algorithm described in [Hentenryck *et al.*, 1992]):

- `composition`$[...](x_i)$: to enforce arc consistency on the lower bound of the domain of x_i, one can simply update this to the position of the first occurrence of the pattern after lb_i in the text t_i. To find this occurrence, the algorithm of Baeza-Yates and Manber [Baeza-Yaltes and Gonnet, 1992; Wu and Manber, 1991] is used. This algorithm is based on a boolean representation of the search state and exploit the intrinsic parallelism of bitwise logical operations in modern CPU. It has a linear complexity for exact string search and a complexity in $O(m \times n)$ for the Levenstein distance (m being the length of the text and n that of the pattern sought). A similar processing can be done on the other bound (but is not used in our prototype).
- `helix`$[...](x_{i_1}, x_{i_2}, x_{i_3}, x_{i_4})$: Consider for example variable x_{i_1}. To filter x_i domain, we must find the first helix (a support) that satisfies the parameters of the constraint. By first we mean the helix with the smallest position of the 5' extremity of the first arm (pointed by x_1). The problem for helices (which can be seen as two related substrings) is more complex than for `composition` since the two strings are initially unknown. This makes it impossible to use string matching algorithms relying on a preprocessing of the string searched. The most naive approach that successively tries all possible positions for the first and second string is obviously quadratic. However, in our case, the distance between the regions where the words may appear is constrained by the length parameters b_{min} and b_{max}. Together with parameters l_{min} and l_{max}, this makes the complexity of the naive approach linear in the text length. This is therefore the method implemented. A similar approach can be used for other bounds.
- `duplex`$[...](x_1, x_2, y_3, y_4)$: this constraint differs from the previous one by the precise fact that there is no

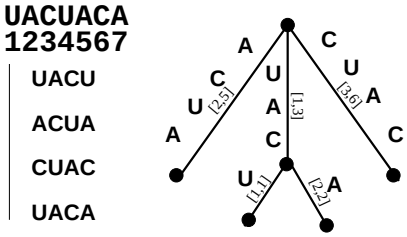

Figure 6: The k factor tree (with $k = 4$) for *UACUACA*. This data structure represents the set of substrings of length 4 of the text.

possible b_{min} and b_{max} parameters since the two interacting substrings do not necessarily appear on the same sequence. The previous naive approach is therefore impractical. We decided to use a specialized version of suffix-trees [McCreight, 1976; Ukkonen, 1992] that captures occurrences of patterns of bounded length. This data structure, called a *k-factor tree* [Allali and Sagot, 2003] allows to perform string search in time linear in the length of the pattern searched (independently of the text length). The data structure, illustrated in Fig. 6, is built once before the search, in space and time linear in the length of the text [McCreight, 1976; Ukkonen, 1992]. The associated filtering algorithm does not enforce generalized bound arc consistency but is only triggered when one of the two variables x_1 or y_3 is assigned. *All* the occurrences of the Watson-Crick reverse complement can then be efficiently found in the k-factor tree and used to update the bounds of the other variables (the position of the first and last possible occurrences define the new bounds).

Because these constraint propagation are quite expensive compared to the simple `distance` constraint, and in order to avoid repeated useless applications of the filtering algorithms, once a support is found it is memorized and will not be sought again until one of its value is deleted (as in AC2001 [Bessiere and Regin, 2001]).

3 Results and discussion

This approach has been implemented in C++ and results in a specific solver called MILPAT: **M**otifs and **I**nter-mo**L**ecular motifs searching tool using cs**P** form**A**lism and solving **T**echniques. We tested our approach on different RNA gene search problems in order to assess its efficiency and modelling capacities.

3.1 tRNAs

The tRNA structure and sequence profiles are perhaps the best studied among RNAs; hence, they are very appropriate for a first benchmarking.

tRNA genes include four helices corresponding respectively to A-stem (7 nucleotide pairs), D-stem (from 3 to 4 nucleotide pairs), C-stem (5 nucleotide pairs) and T-stem (5 nucleotide pairs), six loops corresponding respectively to the single strand between A-stem and D-stem (sequence UN with U invariant), D-loop (4 to 14 nucleotides), the single strand between D-stem and C-stem (one nucleotide), C-loop (6 to 60 nucleotides), the single strand between D-stem and C-stem (one nucleotide), C-loop (6 to 60 nucleotides), the single strand between C-stem and T-stem (also called V-loop, 2 to 22 nucleotides), T-loop (NUC).

Software (genome size)	*E. coli* (4.610^6)	*S. cereviciae* (12.0710^6)
PatScan	1 min. 32	1 h 40
RnaMotif	4 s.	8h40
RnaMot	2 min.	92 h
MILPAT (order A)	39 s	1 h52
MILPAT (order B)	39 s	20 min.

Table 1: Comparison of the time efficiency.

The signature of tRNAs used here is deliberately a simple one that can be modelled in all existing general purpose tools. We have concentrated on finding sequences that can adopt a cloverleaf-like secondary structure within given ranges of stem and loop lengths. We searched the *Escherichia coli* and *Saccharomyces cerevisiae* genomes.

We compared the time execution of MILPAT with three other general purpose programs. The tRNA signature used in our comparisons is from Gautheret and *al.* [Gautheret *et al.*, 1990]. It includes four helices constraints, 14 distance constraints and 2 composition constraints (see Fig. 4). The results of this comparison are shown in Table 1. For each genome search test, all the programs gave the same number of solutions (545 solutions are found for the *E. coli* genome and 849982 for the *S. cerevisiae* genome).

On the computing efficiency basis, three groups may be formed from the slowest to the fastest: (i) RnaMot and RnaMotif, (ii) Patscan and MILPAT with variable selection order A, and (iii) MILPAT with variable selection order B. It is well known that variable assignment order may have a significant influence on efficiency. The static order A used by MILPAT consists in ordering variables according to the topological order of the elements in the structured motif, from 5' to 3'. Order B is a dynamic order following the first fail principle: most constrained variables are chosen first by the backtrack algorithm. Without this order, MILPAT already has an execution time close to the most efficient program, PatScan. Just changing the order leads to an early pruning of the search tree and a considerably improved execution speed for *Saccharomyces cerevisiae*.

3.2 snoRNAs

To validate the ability of MILPAT to model interactions between different molecules, we performed a computational scan of the *Pyrococcus abyssi* genome for C/D snoRNA genes. Since no existing general purpose tool allows to model interaction between a snoRNA and its target, we compared MILPAT to Snoscan, a tailored software for the C/D snoRNA genes. This program sequentially identifies six specific components of these genes (see Fig. 5): a RUGAUGA string (so called C box), a sequence region, able to form a duplex with another "target" sequence and a CUGA string (so called D box). We used a *S. cerevisiae* tailored version of snoScan as no archae-bacteria version is available. This fact probably explains the limited sensitivity shown in Table 2. The descriptor

Software	Solutions	True positives	Time
SnoScan	1611	27	20 min.
MILPAT	852	42	8 s.

Table 2: *Pyrococcus abyssi* genome (1.710^6 characters) - 59 annotated snoRNAs.

used by MILPAT is described in Fig. 5. The missing annotated genes (17 out of 59) are due to the current limitation of the duplex constraint to Watson/Crick matches. These first results show the modelling flexibility and solving efficiency of MILPAT.

4 Conclusion

The main aim of our work is to offer a way of describing new generations of RNA patterns, including the specification of complexes which can be formed by anti-sense interactions between different regions of a genome. The use of CSP methodology together with efficient pattern matching data structures and algorithms provides increased efficiency, extended modelling capabilities for intermolecular interactions and an easily extensible framework.

Beyond this ability to describe inter and intra-molecular interactions with a great flexibility, a number of evolutions are possible to improve MILPAT efficiency and modelling capabilities, including the ability to describe optional or alternative motifs. Within the framework of biological applications, these possibilities are essential to be closer to the structural reality of the molecules.

In its current version, MILPAT is just providing all the true occurrences (satisfying all constraints). It does not optimize any scoring system based on mismatches, thermodynamics or probabilistic parameters. Taking into account such information would require the use of more complex Weighted CSP algorithms such as in [Schiex *et al.*, 1995; Larrosa and Schiex, 2004].

References

[Allali and Sagot, 2003] J. Allali and M. F. Sagot. The at most k-deep factor tree. *Theory of Computer Science*, 2003. in submission.

[Altman *et al.*, 1994] RB Altman, B Weiser, and HF Noller. Constraint satisfaction techniques for modeling large complexes: Application to the central domain of 16s ribosomal rna. In *Proceedings of the second international conference on Intelligent Systems for Molecular Biology*, pages 10–18, 1994.

[Baeza-Yaltes and Gonnet, 1992] R Baeza-Yaltes and GH Gonnet. A new approach to text searching. In *Communications of the ACM*, volume 35, pages 74–82, 1992.

[Barahona and Krippahl, 1999] P Barahona and L Krippahl. Applying constraint programming to protein structure determination. *CP*, pages 289–302, 1999.

[Bessiere and Regin, 2001] C Bessiere and C Regin, J. Refining the basic constraint propagation algorithm. In *IJCAI*, pages 309–315, 2001.

[Billoud *et al.*, 1996] B Billoud, M Kontic, and A Viari. Palingol: a declarative programming language to describe nucleic acids' secondary structures and to scan sequence database. *Nucleic Acids Res*, 24(8):1395–403, 1996.

[Bockhorst and Craven, 2001] J Bockhorst and M Craven. Refining the structure of a stochastic context-free grammar. In *IJCAI*, pages 1315–1322, 2001.

[Dechter, 2003] R Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.

[Dsouza *et al.*, 1997] M Dsouza, N Larsen, and R Overbeek. Searching for patterns in genomic data. *Trends Genet*, 13(12):497–8, 1997.

[Eddy and Durbin, 1994] SR Eddy and R Durbin. Rna sequence analysis using covariance models. *Nucleic Acids Res*, 22(11):2079–88, 1994.

[Eddy, 1996] SR Eddy. Rnabob: a program to search for rna secondary structure motifs in sequence databases. Manual, 1996.

[Eidhammer *et al.*, 2001] I Eidhammer, D Gilbert, I Jonassen, and M Ratnayake. A constraint based structure description language for biosequences. *Constraints*, 6:173–200, 2001.

[Gaspin and Westhof, 1994] C Gaspin and E Westhof. The determination of the secondary structures of RNA as a constraint satisfaction problem. In Frontiers in Artificial Intelligence and Applications. IOS Press, editors, *Advances in Molecular Bioinformatics*. S. Schulze-Kremer, 1994.

[Gautheret *et al.*, 1990] D Gautheret, F Major, and R Cedergren. Pattern searching/alignment with RNA primary and secondary structures: an effective descriptor for trna. *Comput Appl Biosci*, 6(4):325–31, 1990.

[Hentenryck *et al.*, 1992] V Hentenryck, P, Y Deville, and M Teng, C. A generic arc-consistency algorithm and its specializations. *Artif. Intell.*, 57(2-3):291–321, 1992.

[Larrosa and Schiex, 2004] J. Larrosa and T. Schiex. Solving Weighted CSP by maintaining Arc Consistency. *Artificial Intelligence*, 159(1-2):1-26, 2004.

[Lhomme, 1993] O Lhomme. Consistency techniques for numeric CSPs. In *the 13-th International Joint Conference on Artificial Intelligence*, pages 232–238, 1993.

[Macke *et al.*, 2001] TJ Macke, DJ Ecker, RR Gutell, D Gautheret, DA Case, and R Sampath. Rnamotif, an RNA secondary structure definition and search algorithm. *Nucleic Acids Res*, 29(22):4724–35, 2001.

[Major *et al.*, 1991] F Major, M Turcotte, D Gautheret, G Lapalme, E Fillion, and R Cedergren. The combination of symbolic and numerical computation for three-dimensional modeling of RNA. *Science*, 253:1255–1260, 1991.

[McCreight, 1976] EM McCreight. A space-economical suffix tree construction algorithm. *Journal of the ACM*, 23(2):262–272, 1976.

[Muller *et al.*, 1993] G Muller, C Gaspin, A Etienne, and E Westhof. Automatic display of RNA secondary structures. *Cabios*, 9(275):551–561, 1993.

[Policriti *et al.*, 2004] Alberto Policriti, Nicola Vitacolonna, Michele Morgante, and Andrea Zuccolo. Structured motifs search. In *Proceedings of the eighth annual international conference on Computational molecular biology*, pages 133–139. ACM Press, 2004.

[Sakakibara *et al.*, 1994] Y Sakakibara, M Brown, R Hughey, IS Mian, K jölander, RC Underwood, and D Haussler. Recent methods for rna modeling using stochastic context-free grammars. In *CPM '94: Proceedings of the 5th Annual Symposium on Combinatorial Pattern Matching*, pages 289–306. Springer-Verlag, 1994.

[Schiex *et al.*, 1995] T Schiex, H Fargier, and G Verfaillie. Valued Constraint Satisfaction Problems: hard and easy problems. In *Proc. of the International Joint Conference in AI, Montreal, Canada*, 1995.

[Smith and Waterman, 1981] TF Smith and MS Waterman. Identification of common molecular subsequences. *J Mol Biol*, 147(1), 1981.

[Storz, 2002] G Storz. An expanding universe of noncoding rnas. *Science*, 296(5571):1259, 2002.

[Ukkonen, 1992] E Ukkonen. Constructing suffix-trees online in linear time. In *Algorithms, Software, Architecture: Information Processing 92*, pages 484–492, 1992.

[Vialette, 2004] S. Vialette. On the computational complexity of 2-interval pattern matching problems. *Theor. Comput. Sci.*, 312(2-3):223–249, 2004.

[Wu and Manber, 1991] S Wu and U Manber. Fast text searching with errors. Technical report, University of Arizona, 1991.