What Makes the Arc-Preserving Subsequence Problem Hard?

Guillaume Blin, Guillaume Fertin, Romeo Rizzi, Stéphane Vialette

HAL Id: hal-00620359
https://hal.science/hal-00620359

Submitted on 30 Sep 2011

What makes the
Arc-Preserving Subsequence problem hard?[*]

Guillaume Blin[1], Guillaume Fertin[1], Romeo Rizzi[2], and Stéphane Vialette[3]

[1] LINA - FRE CNRS 2729 Université de Nantes,
2 rue de la Houssinière BP 92208 44322 Nantes Cedex 3 - FRANCE
{blin,fertin}@univ-nantes.fr
[2] Università degli Studi di Trento Facoltà di Scienze - Dipartimento di Informatica e Telecomunicazioni
Via Sommarive, 14 - I38050 Povo - Trento (TN) - ITALY
Romeo.Rizzi@unitn.it
[3] LRI - UMR CNRS 8623 Faculté des Sciences d'Orsay, Université Paris-Sud
Bât 490, 91405 Orsay Cedex - FRANCE
vialette@lri.fr

Abstract. Given two arc-annotated sequences (S, P) and (T, Q) representing RNA structures, the Arc-Preserving Subsequence (APS) problem asks whether (T, Q) can be obtained from (S, P) by deleting some of its bases (together with their incident arcs, if any). In previous studies [3, 6], this problem has been naturally divided into subproblems reflecting intrinsic complexity of arc structures. We show that APS(Crossing, Plain) is **NP**-complete, thereby answering an open problem [6]. Furthermore, to get more insight into where actual border of APS hardness is, we refine APS classical subproblems in much the same way as in [11] and give a complete categorization among various restrictions of APS problem complexity.

Keywords: RNA structures, Arc-Preserving Subsequence, Computational complexity.

1 Introduction

At a molecular state, the understanding of biological mechanisms is subordinated to RNA functions discovery and study. Indeed, it is established that the conformation of a single-stranded RNA molecule (a linear sequence composed of ribonucleotides A, U, C and G, also called *primary structure*) partly determines the molecule function. This conformation results from the folding process due to local pairings between complementary bases (A–U and C–G). The RNA secondary structure is a collection of folding patterns that occur in it.

RNA secondary structure comparison is important in many contexts, such as **(i)** identification of highly conserved structures during evolution which suggest a significant common function for the studied RNA molecules [9], **(ii)** RNA classification of various species (phylogeny)[2], **(iii)** RNA folding prediction by considering a set of already known secondary structures [13].

Structure comparison for RNA has thus become a central computational problem bearing many challenging computer science questions. At a theoretical level, RNA structure is often modelled as an *arc-annotated sequence*, that is a pair (S, P) where S is a sequence of ribonucleotides and P represents hydrogen bonds between pairs of elements of S. Different pattern matching and motif search problems have been investigated in the context of arc-annotated sequences among which we can mention Arc-Preserving Subsequence (APS) problem, Edit Distance problem, Arc-Substructure (AST) problem and Longest Arc-Preserving Subsequence (LAPCS) problem (see for instance [3, 8, 7, 6, 1]). For other related studies concerning algorithmic aspects of (protein) structure comparison using *contact maps*, refer to [5, 10].

In this paper, we focus on APS problem: given two arc-annotated sequences (S, P) and (T, Q), this problem asks whether (T, Q) can be exactly obtained from (S, P) by deleting some of its bases together with their incident arcs, if any. This problem is commonly encountered when one is searching for a given RNA pattern in an RNA database [7]. Moreover, from a theoretical point of view, APS problem can be seen as a restricted version of LAPCS problem, and hence has applications in

---

structural comparison of RNA and protein sequences [3, 5, 12]. APS problem has been extensively studied in the past few years [6, 7, 3]. Of course, different restrictions on arc-annotation alter APS computational complexity, and hence this problem has been naturally divided into subproblems reflecting the complexity of the arc structure of both (S, P) and (T, Q): PLAIN, CHAIN, NESTED, CROSSING or UNLIMITED (see Section 2 for details). All of them but one have been classified as to whether they are polynomial time solvable or **NP**-complete. The problem of the existence of a polynomial time algorithm for APS(CROSSING,PLAIN) problem was mentioned in [6] as the last open problem in the context of arc-preserving subsequences. Unfortunately, as we shall prove in Section 4, APS(CROSSING,PLAIN) is **NP**-complete even for restricted special cases.

In analyzing the computational complexity of a problem, we are often trying to define a precise boundary between polynomial and **NP**-complete cases. Therefore, as another step towards establishing the precise complexity landscape of APS problem, we consider that it is of great interest to subdivide existing cases into more precise ones, that is to refine classical complexity levels of APS problem, for determining more precisely what makes the problem hard. For that purpose, we use the framework introduced by Vialette [11] in the context of 2-intervals (a simple abstract structure for modelling RNA secondary structures). As a consequence, the number of complexity levels rises from 4 to 8, and all the entries of this new complexity table need to be filled. Previous known results concerning APS problem, along with our **NP**-completeness proofs, allow us to fill all the entries of this new table, therefore determining what exactly makes the APS problem hard.

The paper is organized as follows. Provided with notations and definitions (Section 2), in Section 3 we introduce and explain new refinements of the complexity levels we are going to study. In Section 4, we show that APS($\{\sqsubset, \between\}, \emptyset$) is **NP**-complete thereby proving that *classical* APS(CROSSING, PLAIN) is **NP**-complete as well. As another refinement to that result, we prove that APS($\{<, \between\}, \emptyset$) is **NP**-complete. Finally, in Section 5, we give new polynomial time solvable algorithms for restricted instances of APS(CROSSING, PLAIN).

2   Preliminaries

An RNA structure is commonly represented as an arc-annotated sequence (S, P) where S is a sequence of ribonucleotides (or bases) and P is a set of arcs connecting pairs of bases in S. Let (S, P) and (T, Q) be two arc-annotated sequences such that $|S| \geq |T|$ (in the following, $n = |S|$ and $m = |T|$). APS problem asks whether (T, Q) can be exactly obtained from (S, P) by deleting some of its bases together with their incident arcs, if any.

Since the general problem is easily seen to be intractable [3], the arc structure must be restricted. Evans [3] proposed four possible restrictions on P (resp. Q) which were largely reused in subsequent literature: (1) there is no base incident to more than one arc, (2) there are no arcs crossing, (3) there is no arc contained in another and (4) there is no arc.

These restrictions are used progressively and inclusively to produce five different levels of allowed arc structure: UNLIMITED - general problem with no restrictions, CROSSING - restriction (1); NESTED - restrictions (1) and (2); CHAIN - restrictions (1), (2) and (3); PLAIN - restriction (4).

Guo proved in [7] that APS(CROSSING, CHAIN) is **NP**-complete. Guo *et al.* observed in [6] that **NP**-completeness of APS(CROSSING, CROSSING) and APS(UNLIMITED, PLAIN) easily follows from results of Evans [3] concerning LAPCS problem. Furthermore, they gave a $O(nm)$ time algorithm for APS(NESTED, NESTED). This algorithm can be applied to easier problems such as APS(NESTED, α) and APS(CHAIN, α) with $\alpha \in \{$CHAIN,PLAIN$\}$. Finally, Guo *et al.* mentioned in [6] that APS(CHAIN, PLAIN) can be solved in $O(n + m)$ time. Observe that UNLIMITED level has no restrictions, and hence is of limited interest in our study. Consequently, from now on we will not be concerned anymore with that level. Until now, the question of the existence of an exact polynomial algorithm for APS(CROSSING, PLAIN) remained open. We will show in the present paper that APS(CROSSING,PLAIN) is **NP**-complete.

3   Refinement of APS problem

In this section, we propose a refinement of APS problem. We first state formally our approach and explain why such a refinement is relevant for both theoretical and experimental studies.

3.1 Splitting the levels.

As we will show soon, APS(CROSSING, PLAIN) is **NP**-complete. That result answers the last open problem concerning APS computational complexity with respect to classical complexity levels. However, we are mainly interested in the elaboration of a precise border between **NP**-complete and polynomially solvable cases. Indeed, both theorists and practitioners might naturally ask for more information concerning APS hard cases in order to get valuable insight into what makes the problem difficult. As a next step towards better understanding what makes APS problem hard, we propose to refine classically models used for classifying arc-annotated sequences. Our refinement consists in splitting those models of arc-annotated sequences into more precise relations between arcs. For example, such a refinement provides a general framework for investigating polynomial time solvable and hard restricted instances of APS(CROSSING, PLAIN), thereby refining in many ways Theorem 1 (see Section 5).

We use three relations first introduced by Vialette [11] in the context of 2-*intervals*. Actually, his definition of 2-intervals could almost apply in this paper (the main difference lies in the fact that 2-intervals are used for representing sets of contiguous arcs). Vialette defined three possible relations between 2-intervals that can be used for arc-annotated sequences as-well. They are the following: for any two arcs $A_1 = (i, j)$ and $A_2 = (k, l)$ in P, we will write $A_1 < A_2$ if $i < j < k < l$ (*precedence* relation), $A_1 \sqsubset A_2$ if $k < i < j < l$ (*nested* relation) and $A_1 \between A_2$ if $i < k < j < l$ (*crossing* relation). Two arcs A_1 and A_2 are τ-comparable for some $\tau \in \{<, \sqsubset, \between\}$ if $A_1 \tau A_2$ or $A_2 \tau A_1$. Let $\mathcal{P}$ be a set of arcs and R be a non-empty subset of $\{<, \sqsubset, \between\}$. The set $\mathcal{P}$ is said to be *R-comparable* if any two distinct arcs of $\mathcal{P}$ are τ-comparable for some $\tau \in R$. An arc-annotated sequence (P, A) is said to be an R-arc-annotated sequence for some non-empty subset R of $\{<, \sqsubset, \between\}$ if A is R-comparable. By abuse of notation, we write $R = \emptyset$ in case $A = \emptyset$. Observe that our model cannot deal with arc-annotated sequences which contain only one arc. However, having only one arc or none can not really affect the problem computational complexity. Just one guess reduces from one case to the other.

As a straightforward illustration of above definitions, classical complexity levels for APS problem can be expressed in terms of combinations of our new relations: PLAIN is fully described by $R = \emptyset$, CHAIN by $R = \{<\}$, NESTED by $R = \{<, \sqsubset\}$ and CROSSING by $R = \{<, \sqsubset, \between\}$. The key point is to observe that our refinement allows us to consider new structures for arc-annotated sequences, namely $R \in \{\{\sqsubset\}, \{\between\}, \{<, \between\}, \{\sqsubset, \between\}\}$, which could not be considered using classical complexity levels. Although other refinements may be possible (in particular well-suited for parameterized complexity analysis), we do believe that such an approach allows a more precise analysis of APS complexity.

Of course one might object that some of these subdivisions are unlikely to appear in RNA secondary structures. However, it is of great interest to answer, at least partly, the following question: Where is the precise boundary between polynomial and **NP**-complete cases ? Indeed, such a question is relevant for both theoretical and experimental studies.

3.2 Immediate results

First, observe that we only have to consider cases of APS(R_1,R_2) where R_1 and R_2 are compatible, *i.e.* $R_2 \subseteq R_1$. Indeed, if this is not the case, we can immediately answer negatively since there exists two arcs in T which satisfy a relation in R_2 which is not in R_1, and hence T simply cannot be obtained from S by deleting bases of S. Those useless cases are simply denoted by hatched areas in Table 1.

Some known results allow us to fill many entries of the new complexity table derived from our refinement. The remainder of this subsection is devoted to detailing these first easy statements. We begin with an easy observation concerning complexity propagation properties of APS problem.

Observation 1 *Let R_1, R_2, R_1' and R_2' be four subsets of $\{<, \sqsubset, \between\}$ such that $R_2' \subseteq R_2 \subseteq R_1$ and $R_2' \subseteq R_1' \subseteq R_1$. If APS($R_1'$, R_2') is **NP**-complete (resp. APS(R_1, R_2) is polynomial time solvable) then so is APS(R_1, R_2) (resp. APS(R_1', R_2')).*

On the positive side, Guo *et al.* have shown that APS(NESTED, NESTED) is solvable in $O(nm)$ time [6]. Another way of stating this is to say that APS($\{<, \sqsubset\}$, $\{<, \sqsubset\}$) is solvable in $O(mn)$ time. That result together with Observation 1 may be summarized by saying that APS(R_1, R_2) for any compatible R_1 and R_2 such that $\lozenge \notin R_1$ and $\lozenge \notin R_2$ is polynomial time solvable.

Conversely, Evans has proved that APS(CROSSING,CROSSING) is **NP**-complete [3]. A simple reading shows that her proof is concerned with $\{<, \sqsubset, \lozenge\}$-arc-annotated sequences, and hence she actually proved that APS($\{<, \sqsubset, \lozenge\}$, $\{<, \sqsubset, \lozenge\}$) is **NP**-complete. Similarly, in proving that APS(CROSSING,CHAIN) is **NP**-complete [7], Guo actually proved that APS($\{<, \sqsubset, \lozenge\}$, $\{<\}$) is **NP**-complete. Note that according to Observation 1, this latter result implies that APS($\{<, \sqsubset, \lozenge\}$, $\{<, \sqsubset\}$) and APS($\{<, \sqsubset, \lozenge\}$, $\{<, \lozenge\}$) are **NP**-complete. Table 1 surveys known and new results for various types of our refined APS problem. Observe that this paper answers all questions concerning APS problem with respect to both classical and new complexity levels.

APS								
$R_1 \setminus R_2$	$\{<, \sqsubset, \lozenge\}$	$\{\sqsubset, \lozenge\}$	$\{<, \lozenge\}$	$\{\lozenge\}$	$\{<, \sqsubset\}$	$\{\sqsubset\}$	$\{<\}$	$\emptyset$
$\{<, \sqsubset, \lozenge\}$	**NP**-C [3]	**NP**-C $\star$	**NP**-C [7]	**NP**-C $\star$	**NP**-C [7]	**NP**-C $\star$	**NP**-C [7]	**NP**-C $\star$
$\{\sqsubset, \lozenge\}$		**NP**-C $\star$	////	**NP**-C $\star$	////	**NP**-C $\star$	////	**NP**-C $\star$
$\{<, \lozenge\}$			**NP**-C $\star$	**NP**-C $\star$	////	////	**NP**-C $\star$	**NP**-C $\star$
$\{\lozenge\}$				$O(nm^2)$ $\star$	////	////	////	$O(nm^2)$ $\star$
$\{<, \sqsubset\}$					$O(nm)$ [6]	$O(nm)$ [6]	$O(nm)$ [6]	$O(nm)$ [6]
$\{\sqsubset\}$						$O(nm)$ [6]	////	$O(nm)$ [6]
$\{<\}$							$O(nm)$ [6]	$O(n+m)$ [6]
$\emptyset$								$O(n+m)$ [6]

Table 1. Complexity results after complexity levels refinement. ////: useless cases. $\star$: results from this paper.

4   Hardness results

We show in this section that APS($\{\sqsubset, \lozenge\}, \emptyset$) is **NP**-complete thereby proving that APS(CROSSING, PLAIN) is **NP**-complete. That result answers an open problem posed by Gramm, Guo and Niedermeier in [6] which is the last open problem concerning the APS computational complexity with respect to classical complexity levels, *i.e.*, PLAIN, CHAIN, NESTED and CROSSING.

We provide a polynomial time reduction from the well known **NP**-complete 3-SAT problem [4]: Given a set $\mathcal{V}_n$ of n variables and a set $\mathcal{C}_q$ of q clauses (each composed of three literals) over $\mathcal{V}_n$, the problem asks to find a truth assignment for $\mathcal{V}_n$ that satisfies all clauses of $\mathcal{C}_q$.

It is easily seen that APS($\{\sqsubset, \lozenge\}, \emptyset$) is in **NP**. The remainder of the section is devoted to proving that it is also **NP**-hard. Let $\mathcal{V}_n = \{x_1, x_2, ... x_n\}$ be a finite set of n variables and $\mathcal{C}_q = \{c_1, c_2, \ldots, c_q\}$ a collection of q clauses. Observe that there is no loss of generality in assuming that literals are left-right ordered, *i.e.*, if $c_i = (x_j \vee x_k \vee x_l)$ then $j < k < l$. Let us first detail the construction of sequences S and T:

$$S = S_{x_1}^s A\ S_{\overline{x_1}}^s\ S_{x_2}^s A\ S_{\overline{x_2}}^s \ldots S_{x_n}^s A\ S_{\overline{x_n}}^s\ S_{c_1}\ S_{c_2} \ldots S_{c_q}\ S_{x_1}^e\ S_{x_2}^e \ldots S_{x_n}^e$$
$$T = T_{x_1}^s\ T_{x_2}^s \ldots T_{x_n}^s\ T_{c_1}\ T_{c_2} \ldots T_{c_q}\ T_{x_1}^e\ T_{x_2}^e \ldots T_{x_n}^e$$

We now detail subsequences composing S and T. Let γ_m (resp. $\gamma_{\overline{m}}$) be the number of occurrences of literal x_m (resp. $\overline{x_m}$) in $\mathcal{C}_q$. For each variable $x_m \in \mathcal{V}_n$, $1 \le m \le n$, we construct words $S_{x_m}^s = AC^{k_m}$, $S_{\overline{x_m}}^s = C^{k_m}A$ and $T_{x_m}^s = AC^{k_m}A$ where C^{k_m} represents a word of $k_m = \max(\gamma_m, \gamma_{\overline{m}})$ consecutive bases C. For each clause c_i of $\mathcal{C}_q$, $1 \le i \le q$, we construct words $S_{c_i} = UGGGA$ and $T_{c_i} = UGA$. Finally, for each variable $x_m \in \mathcal{V}_n$, $1 \le m \le n$, we construct words $S_{x_m}^e = UUA$ and $T_{x_m}^e = UA$.

Having disposed of the two sequences, we now turn to defining the corresponding two arc structures (see Figure 1). In the following, $\mathcal{S}eq[i]$ will denote the ith base of a sequence $\mathcal{S}eq$ and, for any $1 \le m \le n$, $l_{\overline{m}} = |S_{\overline{x_m}}^s|$. For all $1 \le m \le n$, we create the two following arcs: $(S_{x_m}^s[1], S_{x_m}^e[1])$ and $(S_{\overline{x_m}}^s[l_{\overline{m}}], S_{x_m}^e[2])$. For each clause c_i of $\mathcal{C}_q$, $1 \le i \le q$, and for each $1 \le m \le n$, if the kth

(*i.e.* 1st, 2nd or 3rd) literal of c_i is x_m (resp. $\overline{x_m}$) then we create an arc between any free (*i.e.* not already incident to an arc) base C of $S^s_{\overline{x_m}}$ (resp. $S^s_{x_m}$) and the kth base G of S_{c_i} (note that this is possible by definition of $S^s_{\overline{x_m}}$, $S^s_{x_m}$ and S_{c_i}). On the whole, the instance we have constructed is composed of $3q + 2n$ arcs. We denote by APS-CP-construction any construction of this type. In the following, we will distinguish arcs between bases A and U, denoted by AU-arcs, from arcs between bases C and G, denoted by CG-arcs. An illustration of an APS-CP-construction is given in Figure 1. Clearly, our construction can be carried out in polynomial time. Moreover, the result of such a construction is indeed an instance of APS($\{\sqsubset, \emptyset\}, \emptyset$), since $Q = \emptyset$ (no arc is added to T) and P is a $\{\sqsubset, \emptyset\}$-comparable set (since there are no arcs $\{<\}$-comparable).

Lemma 1. *Let (S, P) and (T, Q) be any two arc-annotated sequences obtained from an APS-CP-construction. If (T, Q) can be obtained from (S, P) by deleting some of its bases together with their incident arcs, if any, then for each $1 \leq i \leq q$ and $1 \leq m \leq n$: (i) T_{c_i} is obtained from S_{c_i} by deleting two of its three bases G, (ii) $T^e_{x_m}$ is obtained from $S^e_{x_m}$ by deleting one of its two bases U, (iii) $T^s_{x_m}$ is obtained from $S^s_{x_m} A S^s_{\overline{x_m}}$ by deleting either $S^s_{x_m}$ or $S^s_{\overline{x_m}}$.*

Proof. Let (S, P) and (T, Q) be two arc-annotated sequences resulting from an APS-CP-construction. **(i)** By construction, the first base U appearing in S (resp. T) is $S_{c_1}[1]$ (resp. $T_{c_1}[1]$). Thus, $T_{c_1}[1]$ is obtained from a base U of S at, or after, $S_{c_1}[1]$. Moreover, the number of bases A appearing after $S_{c_1}[1]$ in S is equal to the number of bases A appearing after $T_{c_1}[1]$ in T. Therefore, every base A appearing after $S_{c_1}[1]$ and $T_{c_1}[1]$ must be matched. That is, for each $1 \leq i \leq q$, $T_{c_i}[3]$ is matched to $S_{c_i}[5]$. In particular, $T_{c_q}[3]$ is matched to $S_{c_q}[5]$. But since there are as many bases U between $S_{c_1}[1]$ and $S_{c_q}[5]$ as there are between $T_{c_1}[1]$ and $T_{c_q}[3]$, any base U in this interval in S must be matched to any base U in this interval in T; that is, for any $1 \leq i \leq q$, $T_{c_i}[1]$ is matched to $S_{c_i}[1]$. Thus, we conclude that for any $1 \leq i \leq q$, T_{c_i} is obtained by deleting two of the three bases G of S_{c_i}.
(ii) By the above argument concerning the bases A appearing after $S_{c_1}[1]$ and $T_{c_1}[1]$, we know that if (T, Q) can be obtained from (S, P), then $T^e_{x_m}[2]$ is matched to $S^e_{x_m}[3]$ for any $1 \leq m \leq n$. Thus, for any $1 \leq m \leq n$, $T^e_{x_m}$ is obtained from $S^e_{x_m}$, and in particular $T^e_{x_m}[1]$ is matched to either $S^e_{x_m}[1]$ or $S^e_{x_m}[2]$.
(iii) By definition, as there is no arc incident to bases of T, at least one base incident to every arc of P has to be deleted. We just mentioned that $T^e_{x_m}[1]$ is matched to either $S^e_{x_m}[1]$ or $S^e_{x_m}[2]$ for any $1 \leq m \leq n$. Thus, since by construction there is an arc between $S^e_{x_m}[1]$ and $S^s_{x_m}[1]$ (resp. $S^e_{x_m}[2]$ and $S^s_{\overline{x_m}}[l_{\overline{m}}]$), for any $1 \leq m \leq n$ either $S^s_{x_m}[1]$ or $S^s_{\overline{x_m}}[l_{\overline{m}}]$ has to be deleted; and all these arcs connect a base A appearing before $S_{c_1}[1]$ to a base U appearing after $S_{c_q}[5]$. Therefore, for any $1 \leq m \leq n$ a base A appearing before $S_{c_1}[1]$ in S is deleted. Originally, there are $3n$ bases A appearing before $S_{c_1}[1]$ in S and $2n$ appearing before the first base of $T_{c_1}[1]$ in T. Thus, the number of bases A not deleted in S and appearing before $S_{c_1}[1]$ is equal to the number of bases A appearing before $T_{c_1}[1]$ in T. But since, for each $1 \leq m \leq n$, a base A of either $S^s_{x_m}$ or $S^s_{\overline{x_m}}$ is deleted, we conclude that for each $1 \leq m \leq n$, $T^s_{x_m}$ is obtained from $S^s_{x_m} A S^s_{\overline{x_m}}$, by deleting either $S^s_{x_m}$ or $S^s_{\overline{x_m}}$. □
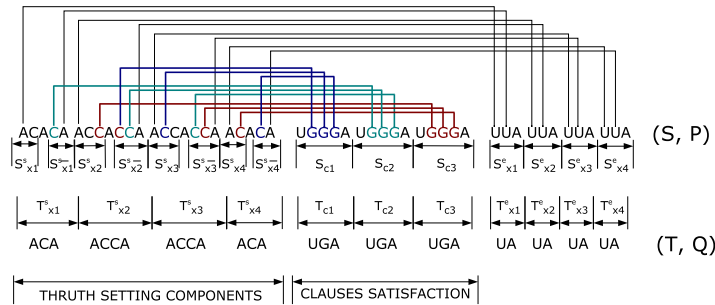

Fig. 1. Example of an APS-CP-construction with $\mathcal{C}_q = (x_2 \vee \overline{x_3} \vee x_4) \wedge (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_2} \vee x_3 \vee \overline{x_4})$.

We now turn to proving that our construction is a polynomial time reduction from 3-SAT to APS(CROSSING, PLAIN).

Lemma 2. *Let I be an instance of* 3-SAT *problem with n variables and q clauses, and I' an instance $((S,P);(T,Q))$ of* APS$(\{\sqsubset, \lozenge\}, \emptyset)$ *obtained by an* APS-CP-*construction from I. An assignment of the variables that satisfies the boolean formula of I exists iff T is an Arc-Preserving Subsequence of S.*

Proof. ($\Rightarrow$) Suppose we have an assignment AS of the n variables that satisfies the boolean formula of I. By definition, for each clause there is at least one literal that satisfies it. In the following, j_i will define, for any $1 \le i \le q$, the smallest index of the literal of c_i (*i.e.* 1, 2 or 3) which, by its assignment, satisfies c_i. Let (S, P) and (T, Q) be two sequences obtained from an APS-CP-construction from I. We look for a set $\mathcal{B}$ of bases to delete from S in order to obtain T.

For each variable $x_m \in AS$ with $1 \le m \le n$, we define $\mathcal{B}$ as follows: **(i)** if $x_m = True$ then $\mathcal{B}$ contains each base of $S^s_{\overline{x_m}}$ and $S^e_{x_m}[1]$, **(ii)** if $x_m = False$ then $\mathcal{B}$ contains each base of $S^s_{x_m}$ and $S^e_{x_m}[2]$, **(iii)** if $j_i = 1$ then $\mathcal{B}$ contains $S_{c_i}[3]$ and $S_{c_i}[4]$, **(iv)** if $j_i = 2$ then $\mathcal{B}$ contains $S_{c_i}[2]$ and $S_{c_i}[4]$, **(v)** if $j_i = 3$ then $\mathcal{B}$ contains $S_{c_i}[2]$ and $S_{c_i}[3]$.

Since a variable has a unique value (*i.e. True* or *False*), either each base of $S^s_{\overline{x_m}}$ and $S^e_{x_m}[1]$ or each base of $S^s_{x_m}$ and $S^e_{x_m}[2]$ are in $\mathcal{B}$ for all $1 \le m \le n$. Thus, $\mathcal{B}$ contains at least one base in S of any AU-arc of P.

For any $1 \le i \le q$, two of the three bases G of S_{c_i} are in $\mathcal{B}$. Thus, $\mathcal{B}$ contains at least one base in S of two thirds of CG-arcs of P. Moreover, $S_{c_i}[j_i + 1]$ is the base G that is not in $\mathcal{B}$. We suppose in the following that the j_i^{th} literal of clause c_i is x_m, with $1 \le m \le n$. Thus, by the way we build the APS-CP-construction, there is an arc between a base C of $S^s_{\overline{x_m}}$ and $S_{c_i}[j_i + 1]$ in P. By definition, if AS is an assignment of the n variables that satisfies the boolean formula, AS satisfies c_i and thus $x_m = True$. We mentioned, in the definition of $\mathcal{B}$ that if $x_m = True$ then each base of $S^s_{\overline{x_m}}$ is in $\mathcal{B}$. Thus, the base C of $S^s_{\overline{x_m}}$ incident to the CG-arc in P with $S_{c_i}[j_i + 1]$ is in $\mathcal{B}$. A similar result can be found if the j_i^{th} literal of clause c_i is $\overline{x_m}$. Thus, $\mathcal{B}$ contains at least one base in S of any CG-arc of P.

If S' is the sequence obtained from S by deleting all the bases of $\mathcal{B}$ together with their incident arcs, if any, then there is no arc in S' (i.e. neither AU-arcs or CG-arcs). By the way we define $\mathcal{B}$, S' is obtained from S by deleting all the bases of either $S^s_{x_m}$ or $S^s_{\overline{x_m}}$, two bases G of S_{c_i} and either $S^e_{x_m}[1]$ or $S^e_{x_m}[2]$, for $1 \le i \le q$ and $1 \le m \le n$. It is easily seen that sequence S' is similar to T.

($\Leftarrow$) Let I be an instance of 3-SAT problem with n variables and q clauses. Let I' be an instance $((S,P);(T,Q))$ of APS$(\{\sqsubset, \lozenge\}, \emptyset)$ obtained by an APS-CP-construction from I such that (T, Q) can be obtained from (S, P) by deleting some of its bases (i.e. a set of bases $\mathcal{B}$) together with their incident arcs, if any. By Lemma 1, either all bases of $S^s_{\overline{x_m}}$ or all bases of $S^s_{x_m}$ are in $\mathcal{B}$. Consequently, for $1 \le m \le n$, we define an assignment AS of the n variables of I as follows: **(i)** if all bases of $S^s_{\overline{x_m}}$ are in $\mathcal{B}$ then $x_m = True$, **(ii)** if all bases of $S^s_{x_m}$ are in $\mathcal{B}$ then $x_m = False$.

Now, let us prove that for any $1 \le i \le q$ clause c_i is satisfied by AS. By Lemma 1, for any $1 \le i \le q$ there is a base G of segment S_{c_i} (say the $j_i + 1^{th}$) that is not in $\mathcal{B}$. By the way we build the APS-CP-construction, there is a CG-arc in P between $S_{c_i}[j_i + 1]$ and a base C of $S^s_{\overline{x_m}}$ (resp. $S^s_{x_m}$) if the j_i^{th} literal of c_i is x_m (resp. $\overline{x_m}$).

Suppose, *w.l.o.g.*, that the j_i^{th} literal of c_i is x_m. Since Q is an empty set, at least one base of any arc of P is in $\mathcal{B}$. Thus, the base C of $S^s_{\overline{x_m}}$ incident to the CG-arc in P with $S_{c_i}[j_i + 1]$ is in $\mathcal{B}$ (since $S_{c_i}[j_i + 1] \notin \mathcal{B}$). Therefore, by Lemma 1, all the bases of $S^s_{\overline{x_m}}$ are in $\mathcal{B}$. By the way we define AS, $x_m = True$ and thus c_i is satisfied. A similar result can be obtained if the j_i^{th} literal of c_i is $\overline{x_m}$. $\qquad\square$

We have thus proved the following.

Theorem 1. APS$(\{\sqsubset, \lozenge\}, \emptyset)$ *is* **NP**-*complete.*

It follows immediately from Theorem 1 that APS$(\{<, \sqsubset, \lozenge\}, \emptyset)$, and hence APS(CROSSING, PLAIN), are **NP**-complete. One might naturally ask for more information concerning hard cases

of APS problem in order to get valuable insight into what makes the problem difficult. Another refinement of the hardness of APS(Crossing,Plain) is given by the following theorem.

Theorem 2. APS($\{<, \lozenge\}, \emptyset$) *is* **NP**-*complete.*

Proof (Sketch of). The proof is also by reduction from 3-Sat. Due to space considerations, the rather technical proof is deferred to the full version of the paper and only sketch main ideas behind Theorem 2. One of the reasons making the proof complicated is that all arcs have to be $\{<, \lozenge\}$-comparable, and hence we can not close an arc before closing all arcs which have been opened before. To overcome this problem we need pairs of strings gadgets which act as "*signal repeaters*" so that we can close and re-open a link carrying a truth value. Actually, repeaters unfortunately invert truth value and hence we need to deal with pairs of repeaters (*first kind* and *second kind* repeaters) where each repeater of the second kind is paired with a repeater of the first kind. □

5 Two polynomial time solvable APS problems

We prove in this section that APS($\{\lozenge\}$, $\emptyset$) and APS($\{\lozenge\},\{\lozenge\}$) are polynomial time solvable. In other words, relation $\lozenge$ alone does not imply **NP**-completeness. We need the following notations. Sequences are the concatenation of zero or more elements from an alphabet. We use the period "." as the concatenation operator, but frequently the two operands are simply put side by side. Let $T = T[1] \, T[2] \dots T[m]$ be a sequence of length m. For all $1 \le i \le j \le m$, we write $T[i:j]$ to denote $T[i] \, T[i+1] \dots T[j]$. The *reverse* of T is the sequence $T^R = T[m] \dots T[2] \, T[1]$. A *factorization* of T is any decomposition $T = x_1 \, x_2 \dots x_q$ where $x_1, x_2, \dots x_q$ are (possibly empty) sequences. Let (T, A) be a $\{\lozenge\}$-arc-annotated sequence and $(i, j) \in A$, $i < j$, be an arc. We call $T[i]$ a *forward base* and $T[j]$ a *backward base*. We will denote by LF_T the position of the last forward base in (T, A) and by FB_T the position of the first backward base in (T, A), *i.e.*, $\mathsf{LF}_T = \max\{i : (i, j) \in A\}$ and $\mathsf{FB}_T = \min\{j : (i, j) \in A\}$. By convention, we let $\mathsf{LF}_T = 0$ and $\mathsf{FB}_T = |T| + 1$ if $A = \emptyset$. Observe that $\mathsf{LF}_T < \mathsf{FB}_T$. We begin by proving a factorization result on $\{\lozenge\}$-arc-annotated sequences.

Lemma 3. *Let S and T be two $\{\lozenge\}$-arc-annotated sequences of length n and m, respectively. If T occurs as an arc preserving subsequence in S, then there exists a factorization (possibly trivial) $T[\mathsf{LF}_T + 1 : \mathsf{FB}_T - 1] = xy$ such that $T[1 : \mathsf{LF}_T] \cdot x \cdot (y \cdot T[\mathsf{FB}_T : m])^R$ occurs as an arc preserving subsequence in $S[1 : \mathsf{FB}_S - 1] \cdot S[\mathsf{FB}_S : n]^R$.*

Proof. Suppose that T occurs as an arc preserving subsequence in S. Since both S and T are $\{\lozenge\}$-arc-annotated sequences, then there exist two factorizations $S[1 : \mathsf{LF}_S] = uw$ and $S[\mathsf{FB}_S : n] = zv$ such that: (i) $T[1 : \mathsf{LF}_T]$ occurs in u, (ii) $T[\mathsf{LF}_T + 1 : \mathsf{FB}_T - 1]$ occurs in $w \cdot S[\mathsf{LF}_S + 1 : \mathsf{FB}_S - 1] \cdot z$ and (iii) $T[\mathsf{FB}_T : m]$ occurs in v. Then it follows that there exists a factorization $T[\mathsf{LF}_T + 1 : \mathsf{FB}_T - 1] = xy$ such that x occurs in $w \cdot S[\mathsf{LF}_S + 1 : \mathsf{FB}_S - 1]$ and y occurs in z, and hence $T' = T[1 : \mathsf{LF}_T] \cdot x \cdot (y \cdot T[\mathsf{FB}_T : m])^R$ occurs as an arc preserving subsequence in $S' = S[1 : \mathsf{FB}_S - 1] \cdot S[\mathsf{FB}_S : n]^R$ □

Theorem 3. APS($\{\lozenge\},\{\lozenge\}$) *is solvable in $O(nm^2)$ time.*

Proof. The algorithm is as follows:

    Data   : Two $\{\lozenge\}$-arc-annotated sequences S and T of length n and m, respectively

    Result : true iff T occurs as an arc-preserving subsequence in S

    begin

1         $S' = S[1 : \mathsf{FB}_S - 1] \cdot S[\mathsf{FB}_S : n]^R$

2   **foreach** *factorization* $T[\mathsf{LF}_T + 1 : \mathsf{FB}_T - 1]| = xy$ **do**

3             $T' = T[1 : \mathsf{LF}_T] \cdot x \cdot (y \cdot T[\mathsf{FB}_T : m])^R$

4   **if** T' *occurs as an arc preserving subsequence in S'* **then**

5   └ **return** *true*

6   **return** *false*

    end

Correctness of the algorithm follows from Lemma 3. What is left is to prove the time complexity. Clearly, $S' = S[1 : \mathsf{FB}_S - 1] \cdot S[\mathsf{FB}_S : n]^R$ is a $\{\sqsubset\}$-arc-annotated sequence. The key point is to note that, for any factorization $T[\mathsf{LF}_T + 1 : \mathsf{FB}_T - 1|] = xy$, the obtained $T' = T[1 : \mathsf{LF}_T] \cdot x \cdot (y \cdot T[\mathsf{FB}_T : m])^R$ is a $\{\sqsubset\}$-arc-annotated sequence as-well. Now let k be the number of arcs in T. So there are at most $m - 2k$ iterations to go before eventually returning false. According to the above, Line 4 constitutes an instance of APS($\{\sqsubset\},\{\sqsubset\}$). But APS($\{\sqsubset\},\{\sqsubset\}$) is a special case of APS($\{<,\sqsubset\},\{<,\sqsubset\}$), and hence is solvable in $O(nm)$ time [6]. Then it follows that the algorithm as a whole runs in $O(nm(m - 2k)) = O(nm^2)$ time. $\qquad\square$

Clearly, the proof of Theorem 3 relies on an efficient algorithm for solving APS($\{\sqsubset\},\{\sqsubset\}$): the better the complexity for APS($\{\sqsubset\},\{\sqsubset\}$), the better the complexity for APS($\{\wr\},\{\wr\}$). We have used only the fact that APS($\{\sqsubset\},\{\sqsubset\}$) is a special case of APS($\{<,\sqsubset\},\{<,\sqsubset\}$). It remains open, however, wether a better complexity can be achieved for APS($\{\sqsubset\},\{\sqsubset\}$). Theorem 3, combined with Observation 1, carries out easily to restricted versions.

Corollary 1. APS($\{\wr\},\emptyset$) *is solvable in* $O(nm^2)$ *time.*

6 Conclusion

In this paper, we investigated the APS problem time complexity and gave a precise characterization of what makes the APS problem hard. We proved that APS(Crossing,Plain) is **NP**-complete thereby answering an open problem posed in [6]. Note that this result answers the last open problem concerning APS computational complexity with respect to classical complexity levels, *i.e.*, Plain, Chain, Nested and Crossing. Also, we refined the four above mentioned levels for exploring the border between polynomial time solvable and **NP**-complete problems. We proved that both APS($\{\sqsubset, \wr\},\emptyset$) and APS($\{<, \wr\},\emptyset$) are **NP**-complete and gave positive results by showing that APS($\{\wr\}, \emptyset$) and APS($\{\wr\},\{\wr\}$) are polynomial time solvable. Hence, the refinement we suggest shows that APS problem becomes hard when one considers sequences containing $\{\wr, \alpha\}$-comparable arcs with $\alpha \neq \emptyset$. Therefore, crossing arcs alone do not imply APS hardness. It is of course a challenging problem to further explore the complexity of the APS problem, and especially the parameterized views, by considering additional parameters such as the cutwidth or the depth of the arc structures.

References

1. J. Alber, J. Gramm, J. Guo, and R. Niedermeier. Computing the similarity of two sequences with nested arc annotations. *Theoretical Computer Science*, 312(2-3):337–358, 2004.
2. G. Caetano-Anollés. Tracing the evolution of RNA structure in ribosomes. *Nucl. Acids. Res.*, 30:2575–2587, 2002.
3. P. Evans. *Algorithms and Complexity for Annotated Sequence Analysis.* PhD thesis, U. Victoria, 1999.
4. M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, 1979.
5. D. Goldman, S. Istrail, and C.H. Papadimitriou. Algorithmic aspects of protein structure similarity. In *Proc. of the 40th Symposium of Foundations of Computer Science (FOCS99)*, pages 512–522, 1999.
6. J. Gramm, J. Guo, and R. Niedermeier. Pattern matching for arc-annotated sequences. In *Proc. of the 22nd Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS02)*, volume 2556 of *LNCS*, pages 182–193, 2002.
7. J. Guo. Exact algorithms for the longest common subsequence problem for arc-annotated sequences. Master's Thesis, Universitat Tubingen, Fed. Rep. of Germany, 2002.
8. T. Jiang, G.-H. Lin, B. Ma, and K. Zhang. The longest common subsequence problem for arc-annotated sequences. In *Proc. 11th Symposium on Combinatorial Pattern Matching (CPM00)*, volume 1848 of *LNCS*, pages 154–165. Springer-Verlag, 2000.
9. V. Juan, C. Crain, and S. Wilson. Evidence for evolutionarily conserved secondary structure in the H19 tumor suppressor RNA. *Nucl. Acids. Res.*, 28:1221–1227, 2000.
10. G. Lancia, R. Carr, B. Walenz, and S. Istrail. 101 optimal PDB structure alignments: a branch-and-cut algorithm for the maximum contact map overlap problem. In *Proceedings of the 5th ACM International Conference on Computational Molecular Biology (RECOMB01)*, pages 193–202, 2001.

11. S. Vialette. On the computational complexity of 2-interval pattern matching. *Theoretical Computer Science*, 312(2-3):223–249, 2004.
12. K. Zhang, L. Wang, and B. Ma. Computing the similarity between RNA structures. In *Proc. 10th Symposium on Combinatorial Pattern Matching (CPM99)*, volume 1645 of *LNCS*, pages 281–293. Springer-Verlag, 1999.
13. M. Zuker. RNA folding. *Meth. Enzymology*, 180:262–288, 1989.