On Bubble Generators in Directed Graphs

Vicente Acuña, Roberto Grossi, Giuseppe Italiano, Leandro Lima, Romeo Rizzi, Gustavo Sacomoto, Marie-France Sagot, Blerina Sinaimeri

Noname manuscript No.
(will be inserted by the editor)

On Bubble Generators in Directed Graphs*

**V. Acuña · R. Grossi · G. F. Italiano ·
L. Lima · R. Rizzi · G. Sacomoto ·
M.-F. Sagot · B. Sinaimeri***

Received: date / Accepted: date

Abstract Bubbles are pairs of internally vertex-disjoint (s, t)-paths in a directed graph, which have many applications in the processing of DNA and RNA data. Listing and analysing all bubbles in a given graph is usually unfeasible in practice, due to the exponential number of bubbles present in real data graphs. In this paper, we propose a notion of bubble generator set, *i.e.,* a polynomial-sized subset of bubbles from which all the other bubbles can be obtained through a suitable application of a specific symmetric difference operator. This set provides a compact representation of the bubble space of

* A shorter version of this paper appeared as the conference article in [1].

* Corrisponding author: Blerina Sinaimeri, E-mail: blerina.sinaimeri@inria.fr

V. Acuña
Center for Mathematical Modeling, Universidad de Chile and UMI CNRS 2807, Santiago, Chile.
E-mail: viacuna@dim.uchile.cl

R. Grossi
Università di Pisa, Pisa, Italy and Erable, INRIA, France.
E-mail: grossi@di.unipi.it

G. F. Italiano
LUISS University, Roma, Italy and Erable, INRIA, France.
E-mail: gitaliano@luiss.it

R. Rizzi
Università di Verona, Verona, Italy.
E-mail: Romeo.Rizzi@univr.it

L. Lima
Erable INRIA Grenoble Rhône-Alpes, Université Lyon 1; CNRS, UMR5558, LBBE, Villeurbanne, France and Università di Roma "Tor Vergata", Roma, Italy.
E-mail: leandro.ishi-soares-de-lima@inria.fr

G. Sacomoto · M.-F. Sagot · B. Sinaimeri
Erable INRIA Grenoble Rhône-Alpes, Université Lyon 1; CNRS, UMR5558, LBBE, Villeurbanne, France.
E-mail: gustavo.sacomoto@gmail.com marie-france.sagot@inria.fr blerina.sinaimeri@inria.fr

a graph. A bubble generator can be useful in practice, since some pertinent information about all the bubbles can be more conveniently extracted from this compact set. We provide a polynomial-time algorithm to decompose any bubble of a graph into the bubbles of such a generator in a tree-like fashion. Finally, we present two applications of the bubble generator on a real RNA-seq dataset.

Keywords Bubbles · Bubble generator set · Decomposition algorithm

1 Introduction

Bubbles are pairs of internally vertex-disjoint (s, t)-paths in a directed graph, which find many applications in the processing of DNA and RNA data. For example, in the genomic context, genome assemblers usually identify and remove bubbles in order to linearise the graph [17, 22, 26]. Bubbles can also represent interesting biological events, *e.g.,* allelic differences (SNPs and indels) when processing DNA data [10, 24, 25], and alternative splicing events in RNA data [19, 18, 13, 20]. Due to their practical relevance, several theoretical studies concerning bubbles were carried out in the past few years [3, 5, 16, 19, 23], usually related to bubble-enumeration algorithms.

Although the enumeration of bubbles could be important to describe biological events appearing in the sequences, this approach has a significant disadvantage. Indeed, while many biological events can be represented by bubbles in a de Bruijn graph (see *e.g.* [20, 15, 18]) (the graph built from the reads provided by a sequencing process), the opposite is not true: most of the bubbles do not correspond to any biological phenomena and appear just because of a combination of other events [13, 18]. In practice, due to the high throughput of second-generation sequencing machines, the genomic and transcriptomic de Bruijn graphs tend to be huge, usually containing from millions to billions of vertices. As expected, the number of bubbles also tends to be huge, in the worst case exponential in the number of vertices. As a consequence, algorithms that deal with bubbles either tend to simplify the graph by removing them, or just enumerate a small subset of the bubbles. Such subsets usually correspond to bubbles with some predefined characteristics, and may not be the best representatives of the biological phenomena under study. More worrying is the fact that, by focusing only on these particular bubbles, all the relevant events described by bubbles that do not satisfy the constraints may be lost. On the other hand, any algorithm that tries to be more exhaustive, say by enumerating a large portion of the bubbles, will certainly spend a prohibitive amount of time in real data graphs and thus it is not likely to be practical [13, 18]. This motivates further work for finding efficient ways to recognise bubbles that correspond to relevant events and/or to represent the set of bubbles in a more concise way.

In this paper, we propose an elementary bubble generator, *i.e.,* a subset of bubbles that is able to generate any other bubble in the graph. More specifically, we show how to identify, for any given directed graph G, a generator set $\mathcal{G}(G)$ of bubbles which is of polynomial size in the input graph, and such that any bubble in G can be obtained in a polynomial number of steps by properly combining the bubbles in the generator $\mathcal{G}(G)$ through a symmetric difference operator. In several biological applications, it is desirable to decompose a bubble into elementary bubbles in such a way that only bubbles can be generated at each step of the decomposition. This happens, for instance, when one wishes to decompose complex alternative splicing events [20] into several elementary alternative splicing events. Our bubble generator enjoys this property: in order to take this into account, we consider a constrained version of the symmetric difference operator, where two bubbles are combinable only if the output is also a bubble (*i.e.,* the operator is undefined if the output is not a bubble). Moreover, we present a $O(n^3)$ time decomposition algorithm that, given a bubble B in the graph G with n vertices, finds a sequence of bubbles from the generator $\mathcal{G}(G)$ whose combination results in B. Our algorithm can be applied when one needs to know how to decompose a bubble into its elementary parts, *e.g.,* when one is interested in identifying and decomposing complex alternative splicing events [20] into several elementary alternative splicing events.

At first sight, a bubble generator might seem related to a cycle basis, which represents a compact description of all Eulerian subgraphs in a graph. The study of cycle bases started a long time ago [14] and has attracted much attention in the last fifteen years, leading to many interesting results, such as the classification of different types of cycle bases, the generalisation of these notions to weighted and to directed graphs, as well as to several complexity results for constructing bases. We refer the interested reader to the books of Deo [7] and Bollobás [4], and to the survey of Kavitha *et al.* [11] for an in-depth coverage of cycle bases. Unfortunately, problems related to bubble generators appear to be very different (and more difficult) from their counterparts in cycle bases, so that it does not seem possible to apply directly to bubble generators all the techniques developed for cycle bases. Indeed, a cycle basis in a directed graph contains subgraphs that are *not* necessarily directed cycles in the original graph, but more generally cycles in the underlying undirected graph [12]. As a consequence, the techniques developed for cycle bases in undirected and directed graphs cannot be applied to our problem, since they do not guarantee a decomposition into elementary bubbles, which generates only bubbles at each step.

To test the practical effectiveness of our generator set of bubbles, we applied it in two different directions in the analysis of a real RNA-seq dataset. First, we consider the use of the generator as a preprocessing step to reduce the graph in input for algorithms that find bubbles, by "cleaning" from the graph all unnecessary arcs (*i.e.* arcs that do not belong to any bubble). Second, we use it to find alternative splicing (henceforth denoted by AS) events in a reference-free context. In particular, some bubbles in our generator set correspond to AS events that are hard to find by the state-of-art algorithm for AS events enumeration [13]. However, this application should still be seen just as a proof-of-concept on the practical potential of the bubble generator or as complementary to current methods, since it is still limited for the exhaustive enumeration of AS events. The latter would require a non-trivial procedure to enumerate AS-associated bubbles by combining generator bubbles and would be beyond the scope of this paper (see Section 6).

The remainder of this paper is organised as follows. Section 2 presents some definitions that will be used throughout the paper. Section 3 introduces our bubble generator. Section 4 presents a $O(n^3)$ time algorithm for decomposing any bubble in a graph into elements of our bubble generator. Section 5 presents two applications of the bubble generator in processing and analysing RNA data. Finally, we conclude with open problems in Section 6.

2 Preliminaries

Throughout the paper we assume that the reader is familiar with the standard graph terminology, as contained for instance in [6]. A graph is a pair $G = (V, E)$, where V is the set of vertices, and $E \subseteq V \times V$ is the set of edges. For convenience, we may also denote the set of vertices V of G by $V(G)$ and its set of edges E by $E(G)$. We further set $n = |V(G)|$ and $m = |E(G)|$. A graph may be *directed* or *undirected*, depending on whether its edges are directed or undirected. In this paper, we will deal with graphs that are directed, unweighted, finite and without parallel edges or self-loops. An edge $e = (u, v)$ is said to be *incident* to the vertices u and v, and u and v are said to be the endpoints of $e = (u, v)$. For a directed graph, edge $e = (u, v)$ is said to be leaving vertex u and entering vertex v. Alternatively, $e = (u, v)$ is an outgoing edge for u and an incoming edge for v. The *in-degree* of a vertex v is given by the number of edges entering v, while the *out-degree* of v is the number of edges leaving v. The *degree* of v is the sum of its in-degree and out-degree.

We say that a graph $G' = (V', E')$ is a *subgraph* of a graph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. Given a subset of vertices $V' \subseteq V$, the subgraph of G *induced* by V', denoted by $G_{V'}$, has V' as vertex set and contains all edges of G that have both endpoints in V'. Given a subset of edges $E' \subseteq E$, the subgraph of G *induced* by E', denoted by $G_{E'}$, has E' as edge set and contains all vertices of G that are endpoints of edges in E'. Given a subset of vertices $V' \subseteq V$ and a subset of edges $E' \subseteq E$, we denote by $G \setminus V'$ the graph induced by $V \setminus V'$ and by $G \setminus E'$ the graph induced by $E \setminus E'$. Given a set S of subgraphs of G, G_S denotes the graph induced by the edges in $\cup_{s \in S} E(s)$. Given two subgraphs G and H, their union $G \cup H$ is the graph F for which $V(F) = V(G) \cup V(H)$ and $E(F) = E(G) \cup E(H)$. Their intersection $G \cap H$ is the graph F for which $V(F) = V(G) \cap V(H)$ and $E(F) = E(G) \cap E(H)$.

Let s, t be any two vertices in G. A (*directed*) *path* from s to t in G is a sequence of vertices and edges $s = v_1$, e_1, v_2, e_2, …, v_{k-1}, e_{k-1}, $v_k = t$, such that $e_i = (v_i, v_{i+1})$ for $i = 1, 2, \ldots, k-1$. Since there is no danger of ambiguity, in the remainder of the paper we will also denote a path simply as $s = v_1$, v_2, …, v_{k-1}, $v_k = t$ (*i.e.,* as a sequence of vertices). A path is *simple* if it does not contain repeated vertices, except possibly for the first and the last vertex.

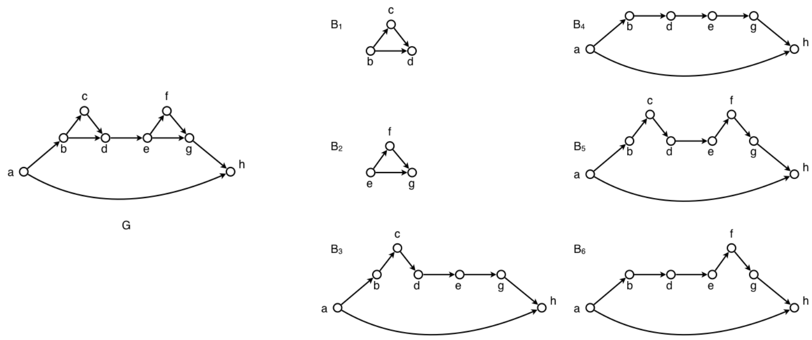

Fig. 1: An example of a graph G and the set $B(G)$ of all the bubbles in G. The set $\mathcal{G}(G) = \{B_1, B_2, B_4\}$ is a generator set that satisfies conditions of Theorem 1.

Throughout this paper, all the paths considered will be simple and referred to as paths. A path from s to t is also referred to as an (s,t)-path. The length of a path p is the number of edges in p and will be denoted by $|p|$. Note that, as a special case, we also allow a single vertex to be a path, *i.e.,* a path of length 0. If p and q are paths, we say that p is a subpath of q if p is contained in q, and we denote this $p \subseteq q$. Given a path p_1 from x to y and a path p_2 from y to z, we denote by $p_1 \cdot p_2$ their concatenation, *i.e.,* the path from x to z defined by the path p_1 followed by p_2. A path q is a prefix of a path p if there exists a path r such that $p = q \cdot r$. Similarly, a path q is a suffix of a path p if there exists a path r such that $p = r \cdot q$. A *(directed) cycle* is a simple path (of length greater than zero) starting and ending on the same vertex.

Definition 1 Given a directed graph G and two (not necessarily distinct) vertices $s, t \in V(G)$, an (s,t)-*bubble* consists of two directed (s,t)-paths that are internally vertex disjoint. Vertex s is the source and t is the target of the bubble. If for a bubble B it holds that $s = t$ then exactly one of the paths of the bubble has length 0, and therefore B corresponds to a directed cycle. In this case, we say that B is a *degenerate bubble*.

In Fig. 1 we show an example of a graph and all the bubbles in it. We denote by $B(G)$ the set of all bubbles in G. Before giving formally the definition of bubble generator of G, we recall some basic definitions of cycle bases in undirected graphs.

Let G be an undirected graph. Two subgraphs G_1, G_2 of G can be combined by the operator $\triangle$ that simply consists in the symmetric difference of the set of edges. More formally, $G_1 \triangle G_2$ is the graph induced by the set of edges$(E(G_1) \cup E(G_2)) \setminus (E(G_1) \cap E(G_2))$. It has been shown that the space of all Eulerian subgraphs of G (called the *cycle space* of G) forms a vector space over $GF(2)$ with the $\triangle$ operation and scalar multiplication $1 \cdot A = A$, $0 \cdot B = \emptyset$ for A, B in the cycle space of G [9,11,12,14]. In the theory of vector spaces, a set of vectors is said to be *linearly dependent* if one of the vectors in the set can be defined as a linear combination of the others; if no vector in the set can be written in this way, then the vectors are said to be *linearly independent* [21]. A *basis* is a minimum set of vectors, such that any vector in the space is a linear combination of this set. Clearly a basis is a set of linearly independent vectors. Furthermore, given a vector space and a set of k linearly independent vectors $\mathcal{F}$, the subspace of vectors generated starting from elements in $\mathcal{F}$ is called the *span* of $\mathcal{F}$ and its dimension is k. It is well-known that a cycle basis for a connected undirected graph G, denoted by $\mathcal{C}(G)$, has dimension $m - n + 1$. If the graph G is not connected this is generalised to $m - n + c$, where c is the number of connected components (see, *e.g.,* [9, 11, 12, 14]) .

As mentioned in Section 1, we are interested in decomposing a bubble into elementary bubbles in such a way that, at each step of the decomposition, only bubbles are generated. To ensure this property, we define next a suitable symmetric difference operator which takes as input two bubbles and produces one bubble as output. Given two bubbles B_1 and B_2, *the constrained symmetric difference* operator $\triangle$ is such that $B_1 \triangle B_2$ is defined if and only if the subgraph induced by $(E(B_1) \cup E(B_2)) \setminus (E(B_1) \cap E(B_2))$ is a bubble. Otherwise, we say that $B_1 \triangle B_2$ is undefined. If $B_1 \triangle B_2$ is defined, we also say that B_1 and B_2 are *combinable*. Given two combinable bubbles B_1 and B_2, we refer to $B_1 \triangle B_2$ as the *sum of* B_1 *and* B_2, and denote it also by $B_1 + B_2$. We also say that the bubble $B_1 + B_2$ is *generated* from bubbles B_1 and B_2, or alternatively that it can be *decomposed* into the bubbles B_1 and B_2. Let $\mathcal{B}$ be a set of bubbles in G. We say that a bubble B is *spanned by* $\mathcal{B}$ if it can be generated starting from bubbles in $\mathcal{B}$. The set of all the bubbles spanned by $\mathcal{B}$ is called the *span of* $\mathcal{B}$. $\mathcal{B}$ is a *bubble generator* if each bubble in G is spanned by $\mathcal{B}$, *i.e.,* each bubble in G can be generated by starting from the bubbles in $\mathcal{B}$.

Due to our constrained symmetric difference operator $\triangle$, all subgraphs generated by the elements in $\mathcal{B}$ are necessarily bubbles. Since not all pairs of bubbles of G are combinable, the bubble space is not closed under $\triangle$, and therefore it does not form a vector space (over $\mathbb{Z}_2$). Hence, the techniques developed for cycle bases cannot be applied directly to bubble generators.

A generator is *minimal* if it does not contain a proper subset that is also a generator; and a generator is *minimum* if it has the minimum cardinality. We are interested in finding a minimum bubble generator of a given directed graph G.

3 The bubble generator

In this section, we present a bubble generator for a directed graph G. Throughout, we assume that shortest paths in G are unique. This is without loss of generality, since there are many standard techniques for achieving this, including perturbing edge weights by infinitesimals. However, for our goal, it suffices to use a "lexicographic ordering". Namely, we define an arbitrary ordering $v_1, \ldots, v_n$ on the vertices of G. A path p is considered lexicographically smaller than a path q if the length of p is strictly smaller than the length of q, or, if p and q have the same length, the sequence of vertices associated with p is lexicographically smaller than the sequence associated with q. We denote this by $p <_{lex} q$.

We denote by $B = (p, q)$ the bubble having p, q as its two internally vertex-disjoint paths, referred to as *legs*. We denote by $\ell(B)$ (resp., by $\mathcal{L}(B)$) the shorter (resp., longer) between the two legs p, q of B. Note that, because of the lexicographic order, there are no ties. We also denote by $|B|$ the number of edges of bubble B. Note that $|B| = |\ell(B)| + |\mathcal{L}(B)|$. Next, we define a total order on the set of bubbles.

Definition 2 Let B_1 and B_2 be any two bubbles. B_1 is *smaller* than B_2 (in symbols, $B_1 < B_2$) if one of the following holds: either (i) $\mathcal{L}(B_1) <_{lex} \mathcal{L}(B_2)$; or (ii) $\mathcal{L}(B_1) = \mathcal{L}(B_2)$ and $\ell(B_1) <_{lex} \ell(B_2)$.

Definition 3 A bubble B is *composed* if it can be obtained as a sum of two smaller bubbles. Otherwise, the bubble B is called *simple*.

For a directed graph G, we denote by $\mathcal{S}(G)$ the set of simple bubbles of G. It is not difficult to see that $\mathcal{S}(G)$ is a generator. We are not able for now to prove that any bubble in G can be obtained in a polynomial number of steps from bubbles in $\mathcal{S}(G)$. Nevertheless, to achieve the latter goal, we will introduce next another generator $\mathcal{G}(G) \supseteq \mathcal{S}(G)$. Let $p : s = x_0, x_1, \ldots, x_h = t$ be a path from s to t and let $0 \leq i \leq j \leq h$. To ease the notation, we denote by $p_{i,j}$ the subpath of p from x_i to x_j, and refer also to $p_{0,j}$ as $p_{s,j}$ and to $p_{i,h}$ as $p_{i,t}$. The next theorem provides some properties of simple bubbles.

Theorem 1 *Let B be a simple (s, t)-bubble in a directed graph G. The following holds:*

(1) $\ell(B)$ is the shortest path from s to t in G;
(2) Let $\mathcal{L}(B) = s, v_1, \ldots, v_r, t$. Then $s, v_1, \ldots, v_r$ is the shortest path from s to v_r in G.

Proof Let B be a simple (s, t)-bubble: we show that both conditions (1) and (2) must hold.

We first consider condition (1). If B is degenerate, then it trivially satisfies condition (1). Therefore, assume that B is non-degenerate and, for contradiction, that $\ell(B)$ is not the shortest path from s to t. Let $p^* : s = x_0, x_1, \ldots, x_h = t$ be the shortest path from s to t in G. For $0 \leq i \leq j \leq h$, by subpath optimality, $p^*_{i,j}$ is the shortest path from x_i to x_j. Let k be the smallest index, $0 \leq k < h$, for which the edge (x_k, x_{k+1}) does not belong to either one of the legs of B. Such an index k must exist, as otherwise p^* would coincide with a leg of B. Furthermore, let l, $k < l \leq h$, be the smallest index greater than k for which $x_l \in V(B)$. Such a vertex x_l must also exist, since $x_h = t \in V(B)$. In other words, x_k is the first vertex of the bubble B where p^* departs from B and x_l, $l > k$, is the first vertex where the shortest path p^* intersects again the bubble B. By definition of x_k and x_l, $p^*_{k,l}$ is internally vertex-disjoint with

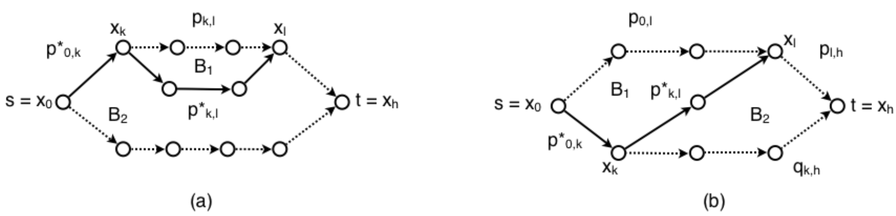

Fig. 2: Case (1) of the proof of Theorem 1. The prefix of the shortest path from s to t is shown as a solid line.

both legs of B. We now claim that B can be obtained as the sum of two smaller bubbles, thus contradicting our assumption that B is a simple bubble.

To prove the claim, we distinguish two cases, depending on whether x_k and x_l are on the same leg of B or not. Consider first the case when x_k and x_l are on the same leg p of B (see Fig. 2(a)). Let B_1 be the bubble with $\ell(B_1) = p^*_{k,l}$ and $\mathcal{L}(B_1) = p_{k,l}$. First, note that if either $x_k \neq s$ or $x_l \neq t$, then $p_{k,l}$ is a proper subpath of a leg of B. Hence, $|\mathcal{L}(B_1)| = |p_{k,l}| < |\mathcal{L}(B)|$, and $B_1 < B$. Otherwise, suppose $s = x_k$ and $t = x_l$. Then either $\mathcal{L}(B_1) = \ell(B) <_{lex} \mathcal{L}(B)$, or $\mathcal{L}(B_1) = \mathcal{L}(B)$ and $\ell(B_1) = p^*_{k,l} = p^* <_{lex} \ell(B)$. In both cases, $B_1 < B$. Let B_2 be the bubble which is obtained from B by replacing $p_{k,l}$ by $p^*_{k,l}$ (see Fig. 2(a)). Since $p^*_{k,l}$ is a shortest path, by subpath optimality, $p^*_{k,l} <_{lex} p_{k,l}$, thus $B_2 < B$. As a result, B can be obtained as the sum of two smaller bubbles B_1, B_2, thus contradicting the assumption that B is simple.

Consider now the case where x_k and x_l are on different legs of B (see Fig. 2(b)). Notice that this means $x_k \neq s$ and $x_l \neq t$. Let p be the leg containing x_l and q the one containing x_k. Note that $p = p_{0,l} \cdot p_{l,h}$ and $q = p^*_{0,k} \cdot q_{k,h}$. Moreover, the two legs of bubble B_1 are $p^*_{0,k} \cdot p^*_{k,l} <_{lex} q$ and $p_{0,l}$, which is a proper subpath of p. Hence, $B_1 < B$. The two legs of bubble B_2 are $q_{k,h}$ which is a proper subpath of q and $p^*_{k,l} \cdot p_{l,h} <_{lex} p$. Hence, $B_2 < B$, and $B = B_1 + B_2$ which implies again that B is not simple.

We show now that B satisfies also condition (2). Assume, for contradiction, that B satisfies condition (1) but not (2), and so $p = s, v_1, \ldots, v_r$ (note that p is equal to $\mathcal{L}(B)$ without its last edge) is not the shortest path from s to v_r in G. Let $p^* : s = x_0, \ldots, x_{h-1} = v_r$, $p^* \neq p$, be such a shortest path in G. Similarly to the previous case, let k be the smallest index, $0 \leq k < h-1$, for which the edge (x_k, x_{k+1}) does not belong to either one of the legs of B, *i.e.* x_k is the first vertex where the shortest path p^* departs from B. Such an index k must exist, as otherwise p^* would be contained in a leg of B. Let $l, k < l \leq h-1$, be the smallest index such that $x_l \in V(B)$. Namely, x_l is the first vertex after x_k where the shortest path p^* intersects again bubble B. Such a vertex x_l must always exist, since $x_{h-1} = v_r \in V(B)$. Since $k < l$, we have that $|p^*_{k,l}| \geq 1$. Furthermore, we claim that x_l must be in $\mathcal{L}(B) \setminus \{s, t\}$. If this were not the case, *i.e.* $x_l \in \ell(B)$, using the assumption that B satisfies condition (1) and hence $\ell(B)$ is a shortest path, we would have two distinct

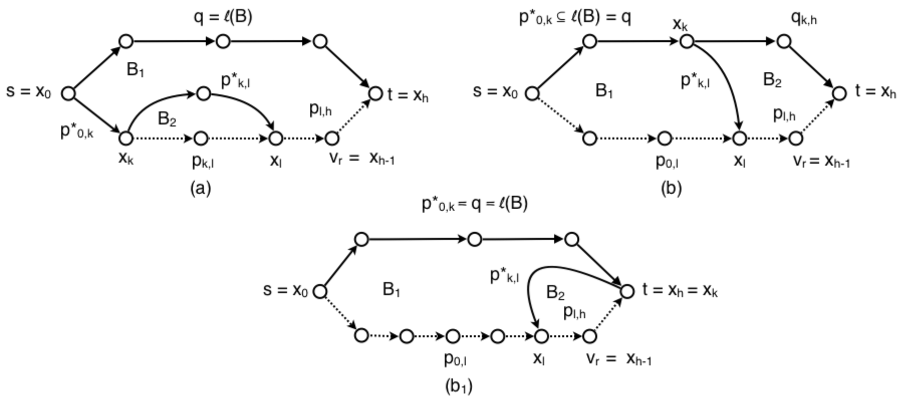

Fig. 3: Case (2) of the proof of Theorem 1. The shortest path from s to t and the prefix of the shortest path from s to v_r are shown as solid lines.

shortest paths from s to x_l in G ($p^*_{x_0,x_l}$ and the subpath of $\ell(B)$ from $s = x_0$ to x_l), which contradicts our assumption that shortest paths are unique.

As $x_l \in \mathcal{L}(B)$ and $x_k \in V(B)$, we need to distinguish two cases: when both x_k, x_l belong to $\mathcal{L}(B)$, and when $x_k \in \ell(B)$ and $x_l \in \mathcal{L}(B)$. We set $p = \mathcal{L}(B), q = \ell(B)$.

In the first case (see Fig. 3(a)), let B_1 be the bubble such that: (a) one leg coincides with $\ell(B)$ and as $\ell(B)$ is the shortest path from s to t and that shortest paths are unique, then this necessarily should be the shorter leg of B_1, hence $\ell(B_1) = \ell(B)$; and (b) the other leg $\mathcal{L}(B_1) = p^*_{0,k} \cdot p^*_{k,l} \cdot p_{l,h}$. Since $p^*_{k,l} <_{lex} p_{k,l}$ then $\mathcal{L}(B_1) <_{lex} \mathcal{L}(B)$, and thus $B_1 < B$. Let B_2 be the bubble with $\ell(B_2) = p^*_{k,l}$, and $\mathcal{L}(B_2) = p_{k,l}$. Since $\mathcal{L}(B_2) \subset \mathcal{L}(B)$ (as $x_k \neq t$), $B_2 < B$. As a result, B can be obtained as the sum of two smaller bubbles B_1, B_2, thus contradicting the assumption that B is simple.

In the second case (see Fig. 3(b)), let B_1 be the bubble with $\ell(B_1) = p^*_{0,k} \cdot p^*_{k,l}$ (notice that $\ell(B_1)$ is indeed the shorter path of B_1 as a subpath of a unique shortest path in the graph) and $\mathcal{L}(B_1) = p_{0,l}$. Since $\mathcal{L}(B_1) \subset \mathcal{L}(B)$, $B_1 < B$. Let B_2 be the bubble with $\ell(B_2) = q_{k,h}$, and $\mathcal{L}(B_2) = p^*_{k,l} \cdot p_{l,h}$. As $\mathcal{L}(B) = p_{0,l} \cdot p_{l,h}$ and $p^*_{0,k} \cdot p^*_{k,l}$ is strictly smaller than $p_{0,l}$, we have $\mathcal{L}(B_2) <_{lex} \mathcal{L}(B)$, $B_2 < B$. Again, B can be obtained as the sum of two smaller bubbles B_1, B_2, thus contradicting the assumption that B is simple. Finally, notice that this includes also the case $x_k = t$ and the argument holds identically with B_2 being a degenerate bubble. For the sake of clarity, we depicted this case separately in Fig. 3(b_1). ■

Given a directed graph G, we denote by $\mathcal{G}(G)$ the set of bubbles in G satisfying conditions (1) and (2) of Theorem 1. An example of a graph together with a generator $\mathcal{G}(G)$ is given in Fig. 1.

Theorem 2 *Let G be a directed graph. The following holds:*

(1) $\mathcal{G}(G)$ is a generator set for all the bubbles of G;
(2) $|\mathcal{G}(G)| \leq nm$.

Proof (1) Recall that $\mathcal{S}(G)$ is the set of simple bubbles. By Theorem 1, $\mathcal{S}(G) \subseteq \mathcal{G}(G)$, and thus $\mathcal{G}(G)$ is a generator set for all the bubbles of G.
(2) Since every bubble b in $\mathcal{G}(G)$, with $\ell(b) = s, u_1, \ldots, t$ and $\mathcal{L}(b) = s, v_1, \ldots, v_r, t$, can be uniquely identified by its vertex s and its edge (v_r, t), then the number of bubbles in $\mathcal{G}(G)$ is upper-bounded by nm. ■

The upper bound given in Theorem 2 is asymptotically tight, as shown by the family of simple directed graphs on vertex set $V_n = \{1, 2, \ldots, n\}$ and all possible $n * (n-1)$ edges in their edge set $E_n = \{(u, v) : u \neq v, u, v \in V\}$.

Remark 1 Conditions (1) and (2) of Theorem 1 are not sufficient to guarantee that a bubble is simple, *e.g.,* see Fig. 4. Thus, the generator $\mathcal{G}(G)$ is not necessarily minimal. Recall that a generator is *minimal* if it does not contain a proper subset that is also a generator; and a generator is *minimum* if it has the minimum cardinality.

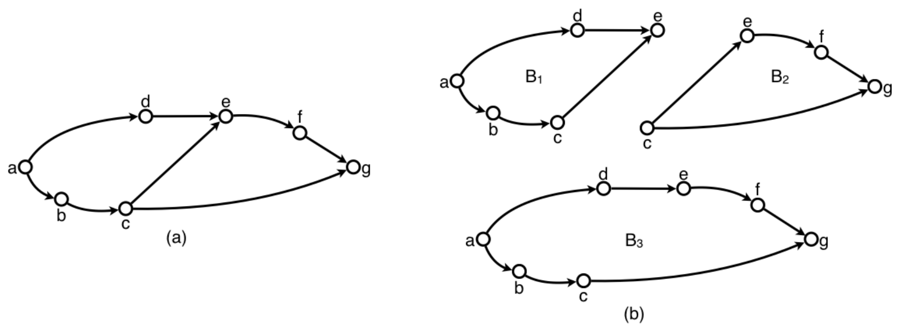

Fig. 4: An example showing that conditions (1) and (2) of Theorem 1 are not sufficient to guarantee that a bubble is simple. (a) A directed graph G. (b) The three bubbles B_1, B_2 and B_3 of G satisfying conditions (1) and (2) of Theorem 1, in which B_1 and B_2 are simple, but B_3 is composed, since $B_1 < B_3$, $B_2 < B_3$ and $B_3 = B_1 + B_2$.

4 A polynomial-time algorithm for decomposing bubbles

The main result of this section is to provide a polynomial-time algorithm for decomposing any bubble B of G into bubbles of $\mathcal{G}(G)$. To do so, we make use of a tree-like decomposition. Intuitively, a bubble B has a tree-like decomposition, if B can be decomposed following a rooted tree structure where each node corresponds to a bubble; in particular, the root and the leaves correspond to B and bubbles in the generator, respectively. Moreover, each bubble in an internal node is obtained by the sum of its children. We need to take extra care in this decomposition since a naive approach could generate (several times) all the bubbles that are smaller than B, yielding an exponential number of steps.

Definition 4 A bubble B is *short* if it satisfies condition (1) of Theorem 1, but not necessarily condition (2). Namely, let $\mathcal{L}(B) = s, v_1, \ldots, v_r, t$ be such that $\ell(B)$ is a shortest path from s to t in G but $s, v_1, \ldots, v_r$ is not necessarily the shortest path from s to v_r in G.

We next introduce a measure for describing how "close" a bubble is to being short.

Definition 5 Given an (s,t)-bubble B, let p^* be the shortest path from s to t. We say that B is k*-short*, for $k \geq 0$, if there is a leg $p \in \{\ell(B), \mathcal{L}(B)\}$ for which p^* and p share a prefix of exactly k edges.

Since in our case shortest paths are unique, only one leg of a bubble B can share a prefix with the shortest path p^*. Furthermore, any bubble B is k-short for some k, $0 \leq k \leq |\ell(B)|$. In particular, a bubble is short if and only if it is k-short for $k = |\ell(B)|$.

Definition 6 Given a k-short bubble, we define the *short residual of* B as follows: $residual_s(B) = |B| - k$.

Since $0 \leq k \leq |\ell(B)|$, and $|B| = |\ell(B)| + |\mathcal{L}(B)|$, we have that $|\mathcal{L}(B)| \leq residual_s(B) \leq |B|$.

We now present our polynomial time algorithm for decomposing a bubble of the graph G into bubbles of $\mathcal{G}(G)$. In the following, we assume that we have done a preprocessing step to compute all-pairs shortest paths in G in $O(mn + n^2 \log n)$ time.

Lemma 1 *Let B be an (s,t)-bubble that is not short. Then, B can be decomposed into two bubbles B_1 and B_2 ($B = B_1 + B_2$), such that: (a) B_1 is short, and (b) $residual_s(B_2) < residual_s(B)$. Moreover, B_1 and B_2 can be found in $O(n)$ time.*

Proof Let B be a k-short (s,t)-bubble, $0 \leq k < |\ell(B)|$ and let $p^* : s = x_0, x_1, \ldots, x_h = t$ be the shortest path from s to t in G. To prove (a), we follow a similar approach to Theorem 1. Since B is k-short, there is a leg $p \in \{\ell(B), \mathcal{L}(B)\}$ such that p^* and p share a prefix of exactly k edges, $0 \leq k < h$. In other terms, leg p starts with edges (x_0, x_1), …, (x_{k-1}, x_k), the edge (x_k, x_{k+1}) is not in leg p, *i.e.*, x_k is the first vertex where the shortest path p^* departs from the leg p. Note that as a special case, $k = 0$ and $x_k = x_0 = s$. Let $l, k < l \leq h$, be the smallest index such that $x_l \in V(B)$. Namely, x_l is the first vertex after x_k where the shortest path p^* intersects again the bubble B. Such a vertex x_l must always exist, since $x_h = t \in V(B)$. Since $k < l$, we have that $|p^*_{k,l}| \geq 1$. We have two possible cases: either the vertices x_k and x_l are on the same leg of B (see Fig. 2(a)) or x_k and x_l are on different legs of B (see Fig. 2(b)). In either case, we can decompose B as $B = B_1 + B_2$, as illustrated in Fig. 2. Note that in both cases, the bubble B_1 is short since one leg of B_1 is a subpath of the shortest path p^*, and hence a shortest path itself by subpath optimality.

Consider now B_2 in Fig. 2. To prove (b), we distinguish among the following three cases: (1) $x_k \neq s$ and vertices x_k and x_l are on the same leg of B; (2) $x_k \neq s$ and vertices x_k and x_l are on different legs of B; (3) $x_k = s$. First, consider case (1) (see Fig. 2(a)) and note that $residual_s(B) = |p_{k,l}| + |p_{l,h}| + |q_{0,h}|$ where q is the other leg of B different from p. Moreover, $residual_s(B_2) = |p_{l,h}| + |q_{0,h}|$. Hence, $residual_s(B) - residual_s(B_2) = |p_{k,l}| \geq |p^*_{k,l}| \geq 1$. Consider now case (2), (see Fig. 2(b)) and note that $residual_s(B) = |p_{0,l}| + |p_{l,h}| + |q_{k,h}|$ and $residual_s(B_2) = |p_{l,h}| + |q_{k,h}|$, and thus $residual_s(B) - residual_s(B_2) = |p_{0,l}| \geq |p^*_{0,k}| + |p^*_{k,l}| \geq 1$. The proof of case (3) is completely analogous to the one of case (1), with $x_k = s$ and $p^*_{0,k} = \emptyset$, and again $residual_s(B) - residual_s(B_2) = |p_{k,l}| \geq |p^*_{k,l}| \geq 1$. In all cases, $residual_s(B) - residual_s(B_2) > 0$, and thus the claim follows. Finally, note that in order to compute B_1 and B_2 from B, it is sufficient to trace the shortest path p^*. Since all shortest paths are pre-computed in a preprocessing step, this can be done in $O(n)$ time. ■

Lemma 2 *Any bubble B can be represented as a sum of $O(n)$ (not necessarily distinct) short bubbles. This decomposition can be found in $O(n^2)$ time in the worst case.*

Proof Each time we apply Lemma 1 to a bubble B, we produce in $O(n)$ time a short bubble B_1 and a bubble B_2 such that $residual_s(B_2) < residual_s(B)$. Since $residual_s(B) \leq |B| \leq n$, the lemma follows. ■

We next show how to further decompose short bubbles. Before doing that, we define the notion of *residual* for short bubbles, which measures how "close" is a short bubble to being a bubble of our generator set $\mathcal{G}(G)$.

Definition 7 Let B be a short (s,t)-bubble, let $\ell(B) = p^*_1$ be the shortest path from s to t in G, and let $\mathcal{L}(B) = s, v_1, \ldots, v_r, t$ be the other leg of B. Let p be the longest prefix of $\mathcal{L}(B) - (v_r, t)$ such that p is a shortest path in G. Then, the *residual of B* is defined as $residual(B) = |\mathcal{L}(B)| - 1 - |p|$.

Since p is a prefix of $\mathcal{L}(B) - (v_r, t)$, we have that $0 \leq |p| \leq |\mathcal{L}(B)| - 1$. Thus, $0 \leq residual(B) \leq |\mathcal{L}(B)| - 1$.

Lemma 3 *Let B be a short (s,t)-bubble such that $residual(B) > 0$. B can be decomposed into two bubbles B_1 and B_2 ($B = B_1 + B_2$) such that B_1 and B_2 are short and $residual(B_1) + residual(B_2) < residual(B)$. Moreover, it is possible to find the bubbles B_1 and B_2 in $O(n)$ time.*

Proof Since B is a short (s,t)-bubble, it satisfies condition (1) of Theorem 1. Furthermore, as $residual(B) > 0$, it does not satisfy condition (2). Therefore, there exists two bubbles $B_1 < B$ and $B_2 < B$ such that $B = B_1 + B_2$ (from Theorem 1). Since $\ell(B)$ is the shortest path from s to t, using arguments similar to the ones in Theorem 1, it can be shown that B can be decomposed into B_1 and B_2 and the only possible cases are the ones depicted in Fig. 3. Note that in all three cases of Fig. 3, each of the bubbles B_1 and B_2 has one leg that is a shortest path. Thus, in all three cases, B_1 and B_2 are short. Moreover, in Fig. 3(a), $residual(B_1) \leq |p_{l,h}| - 1$ and $residual(B_2) \leq |p_{k,l}| - 1$. Therefore, $residual(B_1) + residual(B_2) \leq |p_{l,h}| - 1 + |p_{k,l}| - 1 = residual(B) - 1 < residual(B)$. Similarly, in Fig. 3(b) and (b_1), $residual(B_1) \leq |p_{0,l}| - 1$, $residual(B_2) \leq |p_{l,h}| - 1$, and thus, $residual(B_1) + residual(B_2) \leq |p_{0,l}| - 1 + |p_{l,h}| - 1 = residual(B) - 1 < residual(B)$. In all three cases, B_1 and B_2 are short and $residual(B_1) + residual(B_2) < residual(B)$. The claim thus follows.

Once again, observe that in order to compute B_1 and B_2 from B, it is sufficient to trace the shortest path p^*. Since all shortest paths are pre-computed in a preprocessing step, this can be done in $O(n)$ time. ■

Lemma 4 *Any short bubble B has a tree-like decomposition into $O(n)$ (not necessarily distinct) bubbles from the generator $\mathcal{G}(G)$. This decomposition can be found in $O(n^2)$ time in the worst case.*

Proof Each time we apply Lemma 3 to a short bubble B, we produce in $O(n)$ time two short bubbles B_1 and B_2 such that $residual(B_1) + residual(B_2) < residual(B)$. Since $|\ell(B)| + residual(B) \leq n$, this implies that a short bubble can be decomposed in $O(n)$ bubbles from the generator set $\mathcal{G}(G)$ in $O(n^2)$ time. ■

Theorem 3 *Given a graph G, any bubble B in G can be represented as a sum of $O(n^2)$ bubbles that belong to $\mathcal{G}(G)$. This decomposition can be found in a total of $O(n^3)$ time.*

Proof The theorem follows by Lemma 2 and Lemma 4. ■

5 Applications of the bubble generator in analysing RNA-seq data

In this section, we describe as a proof-of-concept, two applications of the bubble generator to the analysis of RNA-seq data.

Our test dataset is a subset (coming from the same chromosome) of reads of the 58 million RNA-seq Illumina paired-end reads extracted from the mouse brain tissue (available in the ENA repository under the following study: PRJEB25574). The length of the reads is 151bp. We mapped all reads to the *Mus Musculus* reference genome and annotations (Ensembl release 94) using STAR [8]. We then selected only the reads mapping to chromosome 10 of the genome, comprising 4,932,572 reads, as our test dataset. We built the de Bruijn graph from these reads and applied standard sequencing-error-removal procedures, by using KisSplice [13,18], a method to find alternative splicing events in a reference-free context by enumerating bubbles in a de Bruijn Graph. Finally, we extracted the bubble generator from the resulting graph, and evaluated it on two aspects: (i) how well it can preprocess the de Bruijn graph to reduce the work required by a subsequent bubble enumeration algorithm, and (ii) how it performs in terms of finding alternative splicing events. These applications are detailed in the following subsections.

5.1 Preprocessing the de Bruijn graph

Similarly to the practical application of a cycle base, the bubble generator can be used as a preprocessing step in all algorithms that find bubbles, by "cleaning" from the graph all unnecessary edges and vertices, *i.e.* those that do not belong to any bubble. In KisSplice [13,18], this cleaning is based on a biconnected component (BCC) decomposition. A biconnected undirected graph G is a connected graph such that, for any $v \in V(G)$, $G - v$ is connected. Biconnected components (BCCs) are the maximal biconnected subgraphs of a graph G. Given a directed graph, consider its underlying undirected version by ignoring the direction of its edges. Clearly a bubble in the directed graph corresponds to a cycle in the underlying graph, and every edge that belongs to a cycle, belongs also to a BCC of the graph. The graph can then be cleaned by removing every vertex or edge that does not belong to a BCC. This cleaning partitions a potentially massive graph into smaller subgraphs, which are then processed by a bubble enumeration algorithm (*e.g.* [13,18]). However, the BCC-decomposition-based cleaning is not perfect: some vertices and edges might belong only to undirected cycles and not to bubbles.

To improve over this, we perform a more refined cleaning: we compute a bubble generator $\mathcal{G}(G)$ of the directed graph G and we remove every edge and vertex that do not belong to any bubble in $\mathcal{G}(G)$. Notice that this would be a perfect cleaning, meaning that after applying it, every edge of the graph would belong to some bubble.

We evaluated this cleaning procedure on the de Bruijn graph contructed from our test dataset. We first applied the BCC-decomposition-based cleaning on this de Bruijn graph. Then to the result obtained, which is now irreducible by this cleaning, we apply a second cleaning procedure using the bubble generator. The bubble generator cleaning led to a reduction of 40.1% on the number of vertices and of 39.8% on the number of edges. This shows that the generator can indeed yield a better procedure for cleaning the graph, although computing the generator requires more time than computing the BCCs (recall that the BCCs can be computed in linear time). In other words, as expected, a better cleaning comes at the expense of a higher computing time.

5.2 Calling alternative splicing events

As a second application, we consider the problem of finding AS events in a reference-free context. As already mentioned in the introduction, this is a challenging problem in bioinformatics. Indeed, local assemblers such as KisSplice [13] are faced with a dramatically large (and often practically unfeasible) running time due to the exponentially large number of bubbles present, most of which are not interesting as they are not related to AS events. Indeed, a significantly large number of bubbles is due to artifacts of the de Bruijn graph created by repeats longer than the reads (*i.e.,* artificial bubbles not associated with biological events). Hence, in order not to get "lost" in listing false positives, KisSplice relies on heuristics that try to avoid listing bubbles that traverse a repeat-induced subgraph. More specifically, based on the idea that subgraphs of the de Bruijn graph related to repeats have many branching vertices (*i.e.* vertices with in-degree or out-degree at least 2), KisSplice enumerates only bubbles with a number of branching vertices that is below some threshold b. This constraint significantly improved the scalability of KisSplice to the cost of losing the AS events that correspond to bubbles with more than b branching vertices.

The question we tackle in this section is how many AS events we are able to find just by looking at the bubbles in the generator set. Notice that the bubble generator can generate all the bubbles in the graph, thus a first idea is to focus on a subset of it in order to filter out bubbles that are not real AS events. To this purpose, given our dataset we consider the set of bubbles belonging to the generator and the set of bubbles generated by KisSplice (KisSplice being run with default parameters, with a maximum number of branching vertices set to 5). In both cases some simple filters are applied to filter out bubbles that probably do not correspond to AS events (*e.g.* the shorter leg of AS events usually has a length between $2k - 8$ and $2k - 2$, with k being the size of the k-mer in the de Bruijn graph [13,18]). Defining the bubble generator took 716 seconds while KisSplice took 129 seconds. We obtained, as putative AS events, 1403 bubbles for the generator set and 1293 bubbles for KisSplice. In order to assess the precision of our method, we mapped the bubbles output by both methods to the *Mus Musculus* reference genome and annotations (Ensembl release 94) using STAR [8], which were then analysed by KisSplice2RefGenome [2]. KisSplice2RefGenome provides, for each bubble, the gene name, the AS event type (exon skipping, alternative acceptor/donor splice site, intron retention, etc), the genomic coordinates and the list of splice sites used (novel or annotated). We retrieved only those that corresponded to AS events.

Among the generator bubbles classified as putative AS events, 1085 bubbles correspond to true AS events, according to KisSplice2RefGenome, yielding a precision (AS events / putative AS events) of 77.3%. Note that the precision of KisSplice is 90.3% for this dataset. However, what is interesting to see is that 18.5% of the putative AS events from our bubble generator will never be found by KisSplice using the default parameters, as they have more than 5 branching vertices. Moreover, 10% of these bubbles correspond to true AS events that are missed by KisSplice. Increasing the maximum number of allowed branching vertices will increase the running time of KisSplice's algorithm exponentially. A large threshold of b is in practice unfeasible. Since we have bubbles corresponding to putative AS events in the generator that have more than 20 branching vertices, these will be missed by KisSplice.

This analysis shows the practical interest of the bubble generator. Even this simple application led to results that were comparable with the state-of-art algorithm KisSplice and sometimes complementary.

6 Conclusions and open problems

Bubbles in de Bruijn graphs represent interesting biological events, like alternative splicing and allelic differences (SNPs and indels). However, the set of all bubbles in a de Bruijn graph built from real data is usually too large to be efficiently enumerated and analysed. To tackle this issue, in this paper we have proposed a bubble generator, which is a polynomial-sized subset of the bubble space that can be used to generate all and only the bubbles in a directed graph. In particular, we have presented efficient algorithms to identify, for any given directed graph G, a generator set of bubbles $\mathcal{G}(G)$, and to decompose any bubble B in G into bubbles from $\mathcal{G}(G)$. Concerning the applications of the bubble generator, we showed its usefulness in analysing RNA data. In particular, we indicated that our bubble generator can be used in addition to KisSplice to find AS events corresponding to bubbles with a high branching number.

Our work raises several open theoretical questions. First, our generator $\mathcal{G}(G)$ is not necessarily minimal, *i.e.* it might happen that there exists three bubbles $B_1, B_2, B_3 \in \mathcal{G}(G)$ such that $B_1 < B_3$, $B_2 < B_3$, and $B_3 = B_1 + B_2$. Is it possible to find in polynomial time a generator $\mathcal{G}'(G)$ that is minimal? Second, it seems natural to ask whether all minimal generators for bubbles in directed graphs have the same cardinality. Third, it would be interesting to find a generator $\mathcal{G}(G)$ with some additional biologically motivated constraints, as for example the maximum length of the legs of a bubble [19]. Given an integer k and a graph G, is it possible to find a generator $\mathcal{G}(G)$ that generates all and only the bubbles of G which have both legs of length at most k? Fourth, are there faster algorithms to find a bubble generator? Fifth, this work is related to the research done in the direction of cycle bases. However, as we already mentioned, our problem displays characteristics that make it very different from the ones related to cycle bases. Thus, it may be of independent interest to further investigate the connections between those two problems.

There are also some practical questions that need to be addressed in future work, and which might be interesting on their own. We see three possible directions: (i) reduce the false positive AS events by adding more biologically motivated constraints (*e.g.* the ones mentioned in the previous paragraph) to the bubbles in the generator, (ii) find "complex" AS events by listing also the bubbles that result from a combination of two or more bubbles from the generator.

Finally, our polynomial-time decomposition algorithm could be useful in the case where we want to identify and decompose complex alternative splicing events [20] into their elementary parts. We defer all those problems to further investigations.

Acknowledgments

V. Acuña is supported by Fondecyt 1140631, PIA Fellowship AFB170001 and Center for Genome Regulation FONDAP 15090007. R. Grossi and G. F. Italiano are partially supported by MIUR, the Italian Ministry for Education, University and Research, under PRIN Project AHeAD (Efficient Algorithms for HArnessing Networked Data). Part of this work was done while G. F. Italiano was visiting Université de Lyon. L. Lima is supported by the Brazilian Ministry of Science, Technology and Innovation (in portuguese, Ministério da Ciência, Tecnologia e Inovação - MCTI) through the National Counsel of Technological and Scientific Development (in portuguese, Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq), under the Science Without Borders (in portuguese, Ciências Sem Fronteiras) scholarship grant process number 203362/2014-4. B. Sinaimeri, L. Lima and M.-F. Sagot are partially funded by the French ANR project Aster (2016-2020), and together with V. Acuña, also by the Stic AmSud project MAIA (2016-2017). This work was performed using the computing facilities of the CC LBBE/PRABI.

References

1. Acuña, V., Grossi, R., Italiano, G.F., Lima, L., Rizzi, R., Sacomoto, G., Sagot, M., Sinaimeri, B.: On bubble generators in directed graphs. In: Graph-Theoretic Concepts in Computer Science - 43rd International Workshop, WG 2017, Eindhoven, The Netherlands, June 21-23,, *Lecture Notes in Computer Science*, vol. 10520, pp. 18–31. Springer (2017)
2. Benoit-Pilven, C., Marchet, C., Chautard, E., Lima, L., Lambert, M.P., Sacomoto, G., Rey, A., Cologne, A., Terrone, S., Dulaurier, L., Claude, J.B., Bourgeois, C., Auboeuf, D., Lacroix, V.: Complementarity of assembly-first and mapping-first approaches for alternative splicing annotation and differential analysis from RNAseq data. Scientific Reports **8**(1) (2018)
3. Birmelé, E., Crescenzi, P., Ferreira, R., Grossi, R., Lacroix, V., Marino, A., Pisanti, N., Sacomoto, G., Sagot, M.F.: Efficient Bubble Enumeration in Directed Graphs. In: SPIRE, pp. 118–129 (2012)
4. Bollobás, B.: Modern graph theory, *Graduate Texts in Mathematics*, vol. 184. Springer-Verlag, Berlin (1998)
5. Brankovic, L., Iliopoulos, C.S., Kundu, R., Mohamed, M., Pissis, S.P., Vayani, F.: Linear-time superbubble identification algorithm for genome assembly. Theoretical Computer Science **609**, 374–383 (2016)
6. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to Algorithms. The MIT Electrical Engineering and Computer Science Series. MIT Press, Cambridge, MA (1991)
7. Deo, N.: Graph theory with applications to engineering and computer science. Prentice-Hall series in automatic computation. Englewood Cliffs, N.J. Prentice-Hall (1974)

8. Dobin, A., Davis, C.A., Schlesinger, F., Drenkow, J., Zaleski, C., Jha, S., Batut, P., Chaisson, M., Gingeras, T.R.: Star: ultrafast universal rna-seq aligner. Bioinformatics **29**(1), 15–21 (2013)
9. Gleiss, P.M., Leydold, J., Stadler, P.F.: Circuit bases of strongly connected digraphs. Discussiones Mathematicae Graph Theory **23**(2), 241–260 (2003)
10. Iqbal, Z., Caccamo, M., Turner, I., Flicek, P., McVean, G.: De novo assembly and genotyping of variants using colored de bruijn graphs. Nat Genet **44**(2), 226–232 (2012)
11. Kavitha, T., Liebchen, C., Mehlhorn, K., Michail, D., Rizzi, R., Ueckerdt, T., Zweig, K.A.: Cycle bases in graphs characterization, algorithms, complexity, and applications. Computer Science Review **3**(4), 199 – 243 (2009). DOI http://dx.doi.org/10.1016/j.cosrev.2009.08.001
12. Kavitha, T., Mehlhorn, K.: Algorithms to compute minimum cycle bases in directed graphs. Theory of Computing Systems **40**(4), 485 – 505 (2007)
13. Lima, L., Sinaimeri, B., Sacomoto, G., Lopez-Maestre, H., Marchet, C., Miele, V., Sagot, M.F., Lacroix, V.: Playing hide and seek with repeats in local and global de novo transcriptome assembly of short rna-seq reads. Algorithms Mol Biol **12**, 2–2 (2017). DOI 10.1186/s13015-017-0091-2
14. MacLane, S.: A combinatorial condition for planar graphs. Fundamenta Mathematicae **28**, 22–32 (1937)
15. Miller, J.R., Koren, S., Sutton, G.: Assembly algorithms for next-generation sequencing data. Genomics **95**(6), 315–327 (2010)
16. Onodera, T., Sadakane, K., Shibuya, T.: Detecting Superbubbles in Assembly Graphs. In: Algorithms in Bioinformatics, *Lecture Notes in Computer Science*, vol. 8126, pp. 338–348. Springer Berlin Heidelberg (2013)
17. Pevzner, P.A., Tang, H., Tesler, G.: De Novo Repeat Classification and Fragment Assembly. Genome Research **14**(9), 1786–1796 (2004)
18. Sacomoto, G., Kielbassa, J., Chikhi, R., Uricaru, R., Antoniou, P., Sagot, M.F., Peterlongo, P., Lacroix, V.: Kissplice: de-novo calling alternative splicing events from rna-seq data. BMC Bioinformatics **13**(S-6), S5 (2012)
19. Sacomoto, G., Lacroix, V., Sagot, M.F.: A polynomial delay algorithm for the enumeration of bubbles with length constraints in directed graphs and its application to the detection of alternative splicing in RNA-seq data. In: WABI, pp. 99–111 (2013)
20. Sammeth, M.: Complete alternative splicing events are bubbles in splicing graphs. Journal of Computational Biology **16**(8), 1117–1140 (2009)
21. Shilov, G.E.: Linear Algebra. Dover Publications, New York (1977). (Trans. R. A. Silverman)
22. Simpson, J.T., Wong, K., Jackman, S.D., Schein, J.E., Jones, S.J.M., Birol, I.: ABySS: A parallel assembler for short read sequence data. Genome Research **19**(6), 1117–1123 (2009)
23. Sung, W.K., Sadakane, K., Shibuya, T., Belorkar, A., Pyrogova, I.: An $O(m \log m)$-time algorithm for detecting superbubbles. IEEE/ACM Trans. Comput. Biol. Bioinformatics **12**(4), 770–777 (2015)
24. Uricaru, R., Rizk, G., Lacroix, V., Quillery, E., Plantard, O., Chikhi, R., Lemaitre, C., Peterlongo, P.: Reference-free detection of isolated SNPs. Nucleic Acids Research **43**(2), e11 (2015)
25. Younsi, R., MacLean, D.: Using $2k + 2$ bubble searches to find single nucleotide polymorphisms in k-mer graphs. Bioinformatics **31**(5), 642–646 (2015)
26. Zerbino, D., Birney, E.: Velvet: Algorithms for De Novo Short Read Assembly Using De Bruijn Graphs. Genome Res. (2008)