CrossMark

Algorithmic and Hardness Results for the Colorful Components Problems

Anna Adamaszek · Alexandru Popa

Abstract In this paper we investigate the *colorful components* framework, motivated by applications emerging from comparative genomics. The general goal is to remove a collection of edges from an undirected vertex-colored graph G such that in the resulting graph G' all the connected components are *colorful* (i.e., any two vertices of the same color belong to different connected components). We want G' to optimize an objective function, the selection of this function being specific to each problem in the framework. We analyze three objective functions, and thus, three different problems, which are believed to be relevant for the biological applications: minimizing the number of singleton vertices, maximizing the number of edges in the transitive closure, and minimizing the number of connected components. Our main result is a polynomial-time algorithm for the first problem. This result disproves the conjecture of Zheng et al. that the problem is NP-hard (assuming $P \neq NP$). Then, we show that the second problem is APX-hard, thus proving and strengthening the conjecture of Zheng et al. that the problem is NP-hard. Finally, we show that the third problem does not admit polynomial-time approximation within a factor of $|V|^{1/14-\epsilon}$ for any $\epsilon > 0$, assuming $P \neq NP$ (or within a factor of $|V|^{1/2-\epsilon}$, assuming $ZPP \neq NP$).

A preliminary version of this paper has been published in proceedings of LATIN 2014.

A. Adamaszek
Max-Planck-Institut für Informatik, Saarbrücken, Germany
e-mail: anna@mpi-inf.mpg.de

A. Popa (✉)
Faculty of Informatics, Masaryk University, Brno, Czech Republic
e-mail: popa@fi.muni.cz

1 Introduction

In this paper we consider the following framework.

COLORFUL COMPONENTS FRAMEWORK: Given a simple, undirected graph $G = (V, E)$ and a coloring $\sigma : V \rightarrow C$ of the vertices with colors from a given set C, remove a collection of edges $E' \subseteq E$ from the graph such that each connected component in $G' = (V, E \backslash E')$ is a *colorful component* (i.e., it does not contain two identically colored vertices). We want the resulting graph G' to be optimal according to some fixed *optimization measure*.

We consider three optimization measures and, respectively, three different problems: *Minimum Singleton Vertices (MSV)*, *Maximum Edges in Transitive Closure (MEC)*, and *Minimum Colorful Components (MCC)*. We now introduce the optimization measures for all these problems.

Problem 1 (*Minimum singleton vertices*) The goal is to minimize the number of connected components of G' that consist of one vertex.

Problem 2 (*Maximum edges in transitive closure*) The goal is to maximize the number of edges in the transitive closure of G'.

If a graph consists of k connected components, each containing, respectively $a_1, a_2, \ldots, a_k$ vertices, the number of edges in the transitive closure equals

$$\sum_{i=1}^{k} \frac{a_i \cdot (a_i - 1)}{2} \ .$$

Problem 3 (*Minimum colorful components*) The goal is to minimize the number of connected components in G'.

The first two problems have been introduced in [12], while the third one is newly introduced in this paper.

Motivation The colorful components framework is motivated by applications originating from comparative genomics [10,12], which is a fundamental branch of bioinformatics that studies the relationship of the genome structure between different biological species. Research performed in this field can help scientists to improve the understanding of the structure and the functions of human genes and, consequently, find treatments for many diseases [8].

As pointed out in [10,12], one of the key problems in this area, the multiple alignment of gene orders, can be captured as a graph theoretical problem, using the colorful components framework. We refer the reader to [12] for an overview of the connection between the multiple alignment of gene orders and the graph theoretic framework considered, and for a discussion about the biological motivation of two particular problems we consider, MSV and MEC.

Related work We now discuss the collection of known problems which fit into the colorful components framework.

We start with a problem named either *Colorful Components* [4,5] or *Minimum Orthogonal Partition* [7,12], since this problem has received the most attention so far. In this problem the objective function is to minimize the number of edges removed from G to obtain the graph G' in which all the components are colorful. Bruckner et al. show [5] that the problem is NP-hard for three or more colors and they study fixed-parameter algorithms for the problem. Their NP-hardness reduction can be modified slightly (starting the reduction from a version of 3SAT when each variable occurs only $O(1)$ times, instead of from the general 3SAT) to show the APX-hardness of the problem. Zheng et al. [12] and Bruckner et al. [4] study heuristic approaches for the problem, and He et al. [7] present an approximation algorithm for some special case of the problem. As the general problem is a special case of the minimum multi-multiway cut, it admits a $O(\log |C|)$ approximation algorithm [2].

Other objective functions have been proposed, with the hope that some of them are both tractable and biologically meaningful. The MSV and the MEC problems have been introduced by Zheng et al. [12], who presented heuristic algorithms for the problems, without giving any worst-case approximation guarantee. They also conjectured both problems to be NP-hard.

Tremblay-Savard and Swenson [11] consider a Maximum Orthogonal Edge Cover Problem (MAX-OREC), which is a dual problem to MSV. There, the goal is to cover a maximum number of vertices of a graph using vertex-disjoint, non-singleton connected colorful subgraphs. In [11], a 2/3-approximation algorithm for MAX-OREC is presented. Although an approximation algorithm for MAX-OREC does not yield an approximation algorithm for MSV, an optimal solution for MSV gives also an optimal solution for MAX-OREC.

We are not aware of any other results concerning the MSV and MEC problems, or of any previous research on the MCC problem.

Our results Our main result is a polynomial-time *exact* algorithm for the MSV problem, presented in Sect. 2. This disproves the conjecture of Zheng et al. [12] that the problem is NP-hard (assuming $P \neq NP$). Our algorithm maintains a feasible solution $G' = (V, E')$ for the MSV problem, starting with an edgeless graph $G' = (V, \emptyset)$. Then, in each step G' is modified by applying to it a carefully chosen alternating path p, starting at a singleton vertex. The alternating path consists of the edges of G, and its every second edge is in G'. Applying p to G' means that the edges from p which are not in G' are added to G', and at the same time the edges of p which are in G' are removed from G'. The algorithm ensures that at each step G' is a feasible solution to the problem, and satisfies an invariant that all connected components in G' are either singletons, edges or stars. In the analysis we show that when the algorithm does not find any new alternating path, the number of singleton components in G' matches the lower bound presented in Sect. 2.1.

In Sect. 3 we study the MEC problem and we show that the problem is NP-hard and APX-hard when the number of colors in the graph is at least 4. This proves the conjecture of Zheng et al. [12]. We show the result via a reduction from the version of the MAX-3SAT problem where each variable appears at most some constant number of times in the formula (see [1], Section 8.4).

Finally, in Sect. 4 we consider the MCC problem, which is introduced for the first time in this paper. We prove that MCC does not admit polynomial-time approximation within a factor of $|V|^{1/14-\epsilon}$, for any $\epsilon > 0$, unless $P = NP$ (or within a factor of $|V|^{1/2-\epsilon}$, unless $ZPP = NP$), even if each vertex color appears at most two times. We show the inapproximability result via a reduction from Minimum Clique Partition which is equivalent to Minimum Graph Coloring [9].

2 A Polynomial-Time Exact Algorithm for MSV

In this section we present a polynomial-time algorithm MSVEXACT which finds an optimal solution for the MSV problem. First, in Sect. 2.1 we show a lower bound on the number of singleton vertices in any feasible solution for the problem. Then, in Sect. 2.2 we describe the algorithm, with its key procedure presented in Sect. 2.3. The analysis of the algorithm is made in Sect. 2.4.

2.1 Lower Bound

Let a graph $G = (V, E)$, together with a coloring $\sigma : V \to C$, be an instance of the MSV problem. For any color $c \in C$ let $V_c \subseteq V$ denote the set of vertices of color c. For any set of vertices $V' \subseteq V$ we denote by $N(V')$ the set of neighbors of V' in G, i.e. $N(V') = \{v \in V \setminus V' : \exists v' \in V' s.t. (v', v) \in E\}$. For any set of colors $C' \subseteq C$ and set of vertices $V' \subseteq V$ we denote by $N_{C'}(V')$ the set of neighbors of V' in G which have colors in C', i.e. $N_{C'}(V') = \{v \in N(V') : \sigma(v) \in C'\}$.

Lemma 1 *For any color $c \in C$ let*

$$s_c = \max_{V' \subseteq V_c} (|V'| - |N_{C \setminus \{c\}}(V')|) .$$

Then in any feasible solution for MSV there are at least s_c singletons of color c.

Proof Let $G' = (V, E')$, where $E' \subseteq E$, be a feasible solution for G. Fix a color c for which $s_c > 0$ and let $V' \subseteq V_c$ be the subset maximizing the value of s_c. (Notice that s_c depends only on the graph G, and not on G'.) For each vertex $v' \in V'$ which is not a singleton in G' we pick an arbitrary neighbor $n(v')$ in G'. We have $n(v') \in N_{C \setminus \{c\}}(V')$. As any two vertices from V' belong to different connected components in G', the vertices $n(v')$ are pairwise different. The number of vertices of V' which are not singletons in G' is therefore at most $|N_{C \setminus \{c\}}(V')|$. The number of singletons amongst vertices from V', and also the number of singletons of color c, is at least $|V'| - |N_{C \setminus \{c\}}(V')| = s_c$. □

Corollary 1 *Any feasible solution for MSV has at least $\sum_{c \in C} s_c$ singletons.*

2.2 Idea of the Algorithm

We now present an algorithm MSVEXACT which finds an optimal solution for MSV. The input consists of a simple, undirected graph $G = (V, E)$, together with a coloring

---

Input: A simple, undirected graph $G = (V, E)$, a coloring $\sigma : V \to C$
Output: A subgraph of G minimizing the number of singleton vertices, and in which each
        connected component is colorful

1 $G' := (V, \emptyset)$
2 **foreach** $c \in C$ **do**
3   **while** p=ALTERNATING_PATH(G, σ, G', c) is a path **do**
4   apply p to G'
5   **end**
6 **end**

Algorithm 1: MSVEXACT(G, σ)

$\sigma : V \to C$. The algorithm maintains a feasible solution $G' = (V, E')$ (i.e., G' is a subgraph of the input graph G, and every connected component of G' is a colorful component), starting with an edgeless graph $G' = (V, \emptyset)$. In each step the graph G' is modified by applying to it a carefully chosen alternating path p. The alternating path consists of the edges of G, and its every second edge is in G'. Applying p to G' means that the edges from p which are not in G' are added to G', and at the same time the edges of p which are in G' are removed from G'. See Algorithm 1 for the formal description of the algorithm.

The path p is chosen in such a way, that applying it to G' decreases the number of singleton vertices of color c, without increasing the number of singleton vertices of other colors. Additionally, at each step of the algorithm G' satisfies the invariant that each connected component of G' is a singleton, an edge, or a star (where a star is a tree of diameter 2, in particular it has at least 3 vertices).

We will show that when the algorithm stops, i.e., when it does not find any alternating path p which can be applied to G' to decrease the number of singletons of any color, the number of singleton vertices in G' matches the lower bound from Corollary 1.

2.3 Finding an Alternating Path

Let $G' = (V, E')$ be a feasible solution for an instance $(G = (V, E), \sigma)$ of MSV, such that each connected component of G' is a singleton vertex, an edge, or a star. Let $c \in C$ be an arbitrary color, and let $S_c \subseteq V$ be the set of all singletons of color c in G'. We describe a procedure ALTERNATING_PATH(G, σ, G', c) which outputs an alternating path p for G' in G. In the following section we prove that p satisfies all properties outlined in Sect. 2.2, and that when no path is found, the number of singletons of color c in G' matches the lower bound from Lemma 1.

The idea behind the path construction is as follows. We want to find a path starting in some singleton vertex of color c, connecting each vertex of color c with a vertex of color different than c using an edge $e \in E \setminus E'$; and each vertex of color different than c with a vertex of color c using an edge $e \in E'$ (see Fig. 1). We end the construction of the path when the current endpoint $v \notin V_c$ of the path belongs to a connected component of G' to which we can attach an additional vertex of color c (possibly while splitting the component into two parts). Such a case occurs when v is a leaf of a star (which will result in removing v from the star-component and connecting it with
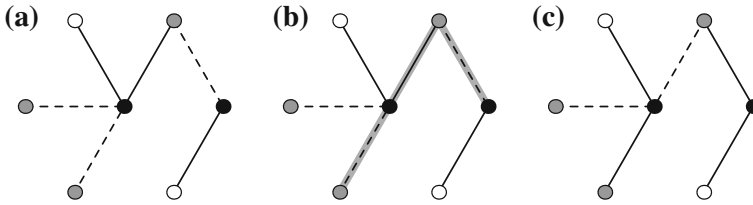
Fig. 1 **a** A subgraph G' (*solid edges*) of G (*dashed and solid edges*) in which every connected component is colorful and it is a singleton, an edge or a star. The vertices are colored with *white*, *gray* and *black*. **b** An alternating path in G' starting in a singleton vertex of color gray. The path ends after reaching a connected component of G' which does not contain a gray vertex. **c** The graph after applying the alternating path. Each colorful component satisfies the conditions stated in **a**, and the number of singletons of color gray decreased

the vertex of color c), or when the connected component of v does not contain color c. Then applying the alternating path to the graph G' results in "switching" vertices of color c between different connected components of G', and removing one singleton of color c, as the start point of the path will not be a singleton in the new graph. The algorithm performs a BFS for the path satisfying the required conditions, starting with the collection of all singleton vertices of color c. See Procedure 2 for a formal description.

Procedure ALTERNATING_PATH constructs the path p as follows. It keeps a set of vertices V' of color c, initially setting $V' := S_c$ (line 1). The set V' will grow over time. Each loop of the algorithm (lines $4 - 18$) considers the set N' of new neighbors in G of the vertices from V' (i.e., the neighbors of V' which have not been

---

Input: A simple, undirected graph $G = (V, E)$, a coloring $\sigma : V \to C$, a feasible subgraph
$\quad\quad G' = (V, E')$ of G, and a color $c \in C$
Output: A path p or NO_PATH_FOUND
1   $V' := S_c$
2   $N' := N_{C \setminus \{c\}}(V')$ `// Neighbors in G`
3   $\forall v \in N' \, \mathrm{pred}(v) := \mathrm{any} v' \in S_c \, \mathrm{s.t.} (v, v') \in E$
4   **while** $|N'| > 0$ **do**
5   **if** $\exists v \in N'$: v *is a leaf of a star in* G' **then**
6          $p :=$ PATH_FROM(v)
7   **return** $p \cup \{(v, v')\}$ s.t. $(v, v') \in E'$
8   **end**
9   **if** $\exists v \in N'$: *the connected component of* v *in* G' *has no color* c **then**
10         $p :=$ PATH_FROM(v)
11   **return** p
12   **end**
13      $V'' := \{v'' \in V_c \setminus V' : \exists v \in N' \text{ s.t. } (v, v'') \in E'\}$
14      $\forall v'' \in V'' \, \mathrm{pred}(v'') := \mathrm{any} v \in N' \mathrm{s.t.} (v, v'') \in E'$
15      $V' := V' \cup V''$
16      $N' := N_{C \setminus \{c\}}(V') \setminus N_{C \setminus \{c\}}(V' \setminus V'')$
17      $\forall v \in N' \mathrm{pred}(v) := \mathrm{any} v' \in V'' \mathrm{s.t.} (v, v') \in E$
18 **end**
19 **return** NO_PATH_FOUND

Procedure 2: ALTERNATING_PATH(G, σ, G', c)

Input: A vertex $v \in V$
Output: An alternating path starting in S_c and ending in v
1 **if** $pred(v) \in S_c$ **then**
2 | **return** (pred(v),v)
3 **end**
4 **return** PATH_FROM(pred(v)) $\cup$ {(pred(v),v)}

<div align="center">

Procedure 3: PATH_FROM(v)

</div>

considered in the previous loops), see lines 2 and 16, in search for vertices which can yield an end of the path (see lines 5, 9). If no such vertex is found, the set V' will be further increased to include the neighbors of N' in G' of color c (line 13, 15). The process continues until an appropriate vertex v is found in N' (lines 5, 9), and then the algorithm returns the alternating path from a singleton of color c to v (see Procedure PATH_FROM(v) invoked in lines 6, 10), or the set N' becomes empty, in which case the answer NO_PATH_FOUND is returned (line 19).

2.4 Analysis

Lemma 2 *When the procedure* ALTERNATING_PATH(G, σ, G', c) *invoked for a graph G' which is a feasible solution for MSV, and s.t. each connected component of G' is a singleton, an edge or a star returns* NO_PATH_FOUND, *then $|S_c| = s_c$.*

Proof If the procedure ALTERNATING_PATH returns NO_PATH_FOUND, then it returns in line 19, i.e., after checking the condition "$|N'| > 0$" (line 4) failed. We show that just before the procedure ends, the following inequality holds:

$$|V'| - |N_{C\setminus\{c\}}(V')| \geq |S_c| \ .$$

If the loop in line 4 has never been entered, we have $V' = S_c$, $N_{C\setminus\{c\}}(V') = N' = \emptyset$, and therefore $|V'| - |N_{C\setminus\{c\}}(V')| = |S_c|$.

Each vertex $v \in N_{C\setminus\{c\}}(V')$ has been inserted into N' at some step of the procedure (line 2 or 16), and subsequently processed in line 5 and 9. As that did not cause the algorithm to return in line 7 or 11, we must have:

– v is not a leaf of a star in G', and
– the connected component containing v contains a vertex colored with c.

As each connected component in G' is a singleton, an edge or a star, and the color of v is different from c, we have two possibilities:

– the connected component of G' containing v is an edge, and the other endpoint of the edge has color c, or
– the connected component of G' containing v is a star containing a vertex of color c, and v is the center of the star.

As any connected component of G' has at most one vertex v satisfying one of the above conditions, any two elements of $N_{C\setminus\{c\}}(V')$ are in different connected components of G'. From the conditions above we also know that each vertex $v \in N_{C\setminus\{c\}}(V')$ has some

neighbor $n(v)$ of color c in G'. Each vertex $n(v)$ has been added to the set V' when the element v has been processed by the procedure (line 13, 15). As any two elements $v_1, v_2 \in N_{C\setminus\{c\}}(V')$ are in different connected components of G', any two vertices $n(v_1), n(v_2)$ are different. As the elements from S_c are singletons in G', and therefore cannot be equal to any $n(v)$, and as $S_c \subseteq V'$, we get $|V'| \geq |S_c| + |N_{C\setminus\{c\}}(V')|$. We obtain the desired inequality.

We have shown that for the set of vertices V' we have $|V'| - |N_{C\setminus\{c\}}(V')| \geq |S_c|$. As $V' \subseteq V_c$, we get $|S_c| \leq \max_{V'' \subseteq V_c}(|V''| - |N_{C\setminus\{c\}}(V'')|) = s_c$. As s_c is a lower bound on $|S_c|$ (see Lemma 1), we get $|S_c| = s_c$.   □

Lemma 3 *Let $G' = (V, E')$ be a feasible solution for MSV s.t. each connected component of G' is a singleton, an edge or a star. Let p be a path returned by* ALTERNATING_PATH(G, σ, G', c), *and let G'' be the result of applying p on G'. Then:*

(a) p is an alternating path for G' in G,

(b) the number of singleton vertices of color c in G'' is smaller than in G'; the number of singleton vertices of any other color does not increase,

(c) each connected component of G'' is a colorful component, and it is a singleton, an edge or a star.

Proof a) First let us show that the procedure PATH_FROM(v) always returns a finite path, and such that the first vertex of p is in S_c. Any vertex which is assigned to the set N' in line 2 is assigned a predecessor from the set S_c (line 3). Any vertex v assigned to N' later, i.e., in some ith iteration of the loop (line 16), is assigned a predecessor $\mathrm{pred}(v) \in V'$, and such that $\mathrm{pred}(v)$ entered V' in the same ith iteration of the loop. Any vertex $v \in V' \setminus S_c$ enters the set V' in some ith iteration of the loop (line 13, 15), and then is assigned a predecessor $\mathrm{pred}(v) \in N'$, such that $\mathrm{pred}(v)$ has been assigned to N' in the previous iteration of the loop (or in line 2, in case $i = 1$). This shows that the procedure PATH_FROM(v) does not loop, and it will eventually (i.e., after at most $|V|$ steps) find a beginning of a path, which is a vertex from the set S_c.

We will now show that every odd vertex of the path p is in V_c (except possibly of the last vertex of p, if it is the vertex v' appended to the path directly by the procedure in line 7), and every even vertex is in $V \setminus V_c$. We already know that the first vertex of the path is in $S_c \subseteq V_c$. As all vertices from the set V' have color c, and all vertices from the set N' have color different from c, a predecessor of a vertex from V' is in N' (line 14) and a predecessor of a vertex from N' is in V' (line 17), the claim follows. Notice that if the procedure reaches line 7, then there are no color requirements for the last vertex v' appended at the end of the path directly, and not using the procedure PATH_FROM (line 7).

We will now show that every even edge of the path p is in E', and every odd edge of p is in $E \setminus E'$, which will prove that p is an alternating path. Let us consider even edges first. An even edge is an edge between some odd vertex v and a preceding vertex w. There can be two cases, and for both of them we obtain that the edge is in E':

- v is a vertex appended to the path directly by the procedure in line 7. Then the edge connecting v with the preceding vertex w is in E' (see line 7).
- v has been appended to the path by the procedure PATH_FROM. Then $w = \mathrm{pred}(v)$ and, from the paragraph above, $v \in V_c$. A vertex from V_c is connected with its predecessor via an edge in E' (see line 14).

Now let us consider odd edges. As the path p starts in a singleton vertex of G', the first (odd) edge of the path is not in E'. Let (v', v) be any other odd edge of p. We have $v' = \text{pred}(v)$, $v' \in V_c$. Let $w = \text{pred}(v')$. From the reasoning above we have $(w, v') \in E'$, as an even edge of p. As w has been processed by the procedure in an earlier loop than v (see the first paragraph of the proof), and processing w did not cause the procedure to return in line 7 or 11, one of the following holds (as each connected component of G' is a singleton, an edge or a star):

– the connected component of G' containing w is an edge, i.e., it is exactly the edge (w, v'), or
– the connected component of G' containing w is a star with center w, in which case v' is a leaf of this star.

In both cases v' has degree one is G', either as an endpoint of an edge (w, v'), or as a leaf of a star with center w, and so $(v', v) \notin E'$.

Notice that from the reasoning above we obtain that for every even edge (w, v') of the path (i.e., for an edge of p which is in E') which is not the last edge of p, the connected component of G' containing (w, v') is either an edge or a star with center w (where $w = \text{pred}(v')$). We will use that later in the proof of c).

See Fig. 2 for an example of an alternating path p for G' in G, and the result of applying this path.

b) From a) we know that the path p applied to G' to construct G'' is an alternating path, i.e., after applying it the degree of each vertex other than the endpoints of the path does not change. Therefore the only vertices of the graph which could become or stop being singletons are the endpoints of p.

The path starts with a vertex $v \in S_c$ (see a)), which is a singleton in G', and therefore the first edge of p is not in E'. The first edge of p is added to the graph and
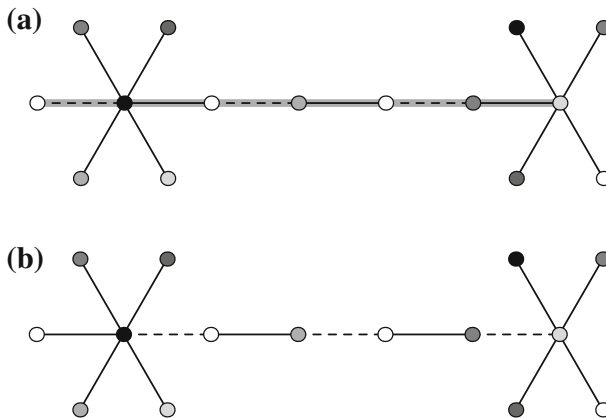
Fig. 2 **a** An alternating path p (*gray edges*) for a subgraph $G' = (V, E')$ (*solid edges*) of $G = (V, E)$ (*dashed and solid edges*). The white vertices denote the vertices of the currently considered color c. The path starts with an edge of $E \setminus E'$, and alternates between the edges of E and $E \setminus E'$. Every second vertex of the path, except of the last vertex, is white. **b** Graph G'' (*solid edges*) obtained from G' by applying the alternating path p. All connected components of G'' are colorful, and are singletons, edges or stars

v stops being a singleton vertex. That decreases the number of singleton vertices of color c by one.

We now have to consider the last vertex of the path. Again, if the last edge is not in E' then the endpoint of the path gets one additional edge incident with it, and so it cannot become a singleton. The only possibility when the last edge of p is in E' is when the path has an even number of edges (as it is an alternating path starting with an edge in $E \setminus E'$), i.e., it ends with an odd vertex. The procedure PATH_FROM is always invoked for a vertex $v \in N'$ (line 6,10) and the path returned by it has an odd number of edges (see point a), where we show that such path alternates between vertices from V' and N'. The only possibility that the path has an even number of edges is when it is generated in line 7, when an additional edge (v, v') is appended at the end of the path. Then the edge (v, v') is removed from G'' and the degree of v' drops by one. However, from line 5 we get that then v is a leaf of a star, and as $(v, v') \in E'$ we have that v' is a center of a star. The degree of v' in G' is at least 2, so v' does not become a singleton after applying the path p to G'.

c) Every connected component of G' is colorful and is either a singleton, an edge or a path. Removing edges does not destroy these properties, therefore it is enough to consider components of G'' which contain some newly added edge. Let $(u, v) \in E$ be an edge added to G'', i.e., an edge of p which is in $E \setminus E'$. From the discussion in a) we know that (u, v) is then an odd edge of the path, and it connects an even vertex $v \in V \setminus V_c$ with its predecessor $u =$pred$(v) \in V_c$. It is enough to show that the connected component of G'' containing the edge (u, v) is either an edge or a star with center v, and that it is colorful.

First, we will show that the degree of u in G'' is one. If u is the start of the path, it has degree 0 in G' and therefore degree 1 in G''. Otherwise, from the reasoning in a) we know that the connected component of G' which contains u is either an edge $($pred$(u), u)$ or a star with a center pred(u). In both cases the edge $($pred$(u), u) \in p$ and it will be removed by applying p. The vertex u will have degree 1 in G''.

We will now consider the following cases. If the construction of the path p finished after processing the vertex v in line 5 (see Fig. 2a when (u, v) is the last dashed line of the path), then v is a leaf of a star in G', and the edge (v, w) connecting v with the center of the star w is in p. Then the edge (v, w) will be removed and the connected component of G'' containing the edge (u, v) will consist only of the edge (u, v). The component is colorful.

If the construction of the path p finished after processing the vertex v in line 9, then the connected component of G' containing v does not contain color c and it is either a singleton, an edge or a star with center v. The path p ends at v. In all the cases, after adding the edge (u, v) the component containing v will remain colorful and it will be an edge or a star.

If the construction of the path p continued after processing the vertex v (see Fig. 2a when (u, v) is the first or the second dashed line of the path), the connected component of G' containing v is an edge or a star with center v, and the component contains a vertex $w \in V_c$. Then w is the successor of v on p, and the edge (v, w) will be removed from the graph when applying p. The connected component containing the edge (u, v) will be an edge or a star with center v, and it will remain colorful. □

Using Lemmas 2 and 3 we can show the main result of this section.

Theorem 1 *The algorithm* MSVEXACT(G, σ) *finds an optimal solution for the MSV problem in time* $O(|V| \cdot |E|)$.

Proof The algorithm MSVEXACT(G, σ) starts by choosing a feasible solution $G' = (V, \emptyset)$ for the problem, in which every connected component is a singleton. Lemma 3 implies that after each step of executing the procedure ALTERNATING_PATH(G, σ, G', c), the new graph G'' obtained is a feasible solution which is a collection of singletons, edges and stars (Lemma 3c). As in each step where finding an alternating path has been successful the number of singleton vertices of the currently processed color $c \in C$ decreases, and for other colors it does not increase (Lemma 3b), after $O(|V|)$ steps the algorithm does not find any more alternating paths. Thus, as each color $c \in C$ has been processed by the algorithm, from Lemma 2 for each color $c \in C$ the number of singleton vertices of color c equals s_c and the resulting graph G' is an optimal solution to the MSV problem (see Corollary 1). As each execution of the procedure ALTERNATING_PATH takes $O(|E|)$ time, the running time of the algorithm MSVEXACT(G, σ) is $O(|V| \cdot |E|)$. □

3 Hardness of MEC

In this section we prove the NP-hardness and the APX-hardness of the MEC problem, for $|C| \geq 4$. We show our result via a reduction from MAX-3SAT(β), a version of the MAX-3SAT problem where each variable appears at most β times in the formula. For $\beta = 3$ the problem is APX-hard (see [1], Section 8.4).

3.1 Reduction from MAX-3SAT (β)

Given an instance of the MAX-3SAT(β) problem, i.e., a 3-CNF formula ϕ with m clauses and n variables, where each variable appears at most β times, we construct an instance of the MEC problem. Our instance is a vertex colored graph $G = (V, E)$, where the vertices are colored with colors from a four-element set $\{a, b, c, v\}$. An example of the reduction is illustrated in Fig. 3.
First we describe the set of vertices V.

1. We add to V a set of vertices $c_1, \ldots, c_m$, each colored with color c, where vertex c_i corresponds to the ith clause of the formula.
2. For a variable x, let n_x be the number of occurrences of the literals x and $\neg x$ in ϕ. For each variable x, we add to V: n_x vertices of color a (denoted by $a_1^x, a_2^x, \ldots, a_{n_x}^x$), n_x vertices of color b (denoted by $b_1^x, b_2^x, \ldots, b_{n_x}^x$), and $2n_x$ vertices of color v (denoted by $v_1^x, v_2^x, \ldots, v_{n_x}^x$ and $w_1^x, w_2^x, \ldots, w_{n_x}^x$). Intuitively, the vertices v_i^x and w_i^x are associated with x and $\neg x$, respectively.

We now show how to construct the set of edges E.

1. For each variable x, we construct a cycle of length $4n_x$ by adding to E the collection of edges (a_i^x, v_i^x), (v_i^x, b_i^x), (b_i^x, w_i^x) and $(w_i^x, a_{(i \bmod n_x)+1}^x)$ for $i = 1, \ldots, n_x$.
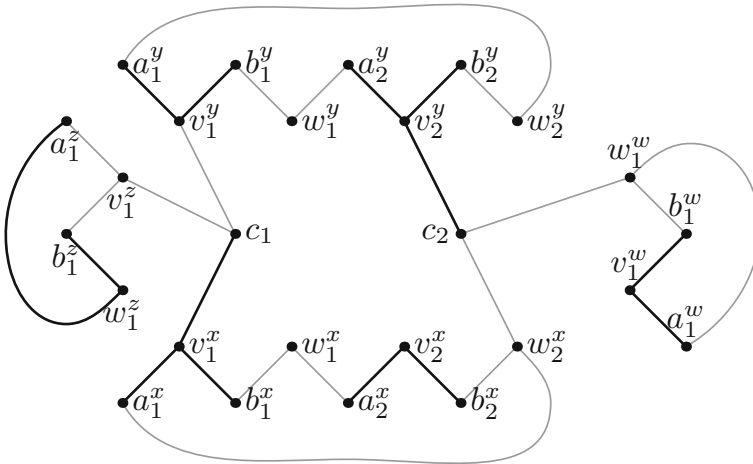
Fig. 3 An instance G of the MEC problem corresponding to the 3SAT formula $(x \vee y \vee z) \wedge (\neg x \vee y \vee \neg w)$ (*both black and gray edges*). The subgraph G'' consisting of all vertices and only black edges represents a solution for G corresponding to the following assignment: $f(x) = f(y) = f(w) = \text{TRUE}$, $f(z) = \text{FALSE}$

2. For each clause we add to E three edges, where each edge connects the vertex c_i representing the clause with a vertex representing one literal of c_i. More formally, if a literal x ($\neg x$) occurs in the ith clause, we add to E an edge connecting c_i with some vertex v_j^x (w_j^x, respectively). We do this operation in such a way, that each vertex v_j^x and w_j^x representing a literal is incident with at most one clause-vertex c_i. Notice that since we have more vertices v_j^x and w_j^x than actual literals, some of the vertices v_j^x and w_j^x will not be connected with any clause-vertex c_i.

3.2 Analysis of the Reduction

Let ϕ be a MAX-3SAT(β) formula on m clauses, and $G = (V, E)$ a vertex-colored graph obtained from ϕ by our reduction. Let $G' = (V, E')$ be a subgraph of G which is an optimal solution for the MEC problem on G.

Lemma 4 *If the formula ϕ is satisfiable, then the transitive closure of G' has at least 12m edges.*[1]

Proof We construct a graph $G'' = (V, E'')$ which is a subgraph of G in the following way (see Fig. 3). Fix a satisfying assignment f for ϕ. For each clause, represented by a vertex c_i, we choose arbitrarily a literal x ($\neg x$) which is satisfied by the assignment f. Let v_j^x (w_j^x, respectively) be the vertex corresponding to the chosen literal which is incident with c_i in G. We add the edge (c_i, v_j^x) ((c_i, w_j^x), respectively) to G''.

---

[1] It can be proven that in this case the transitive closure of G' has exactly $12m$ edges, but that is not needed in the later part of the reasoning.

Additionally, each vertex v_j^x and w_j^x associated with a literal satisfied by f is connected in G'' with the neighboring vertices of color a and b.

It is straightforward to check that G'' is a feasible solution for the MEC problem (i.e., each connected component of G'' is colorful), and that G'' has m connected components containing 4 vertices, $2m$ connected components containing 3 vertices, and $3m$ singletons. The transitive closure of G'' has $6 \cdot m + 3 \cdot 2m = 12m$ edges. As G' is an optimal solution for the MEC problem in G, the transitive closure of G' has at least as many edges as the transitive closure of G''. □

Lemma 5 *If any assignment can satisfy at most a $(1 - \epsilon)$ fraction of the m clauses of the formula ϕ, then the transitive closure of G' has at most $12m - \Theta(\epsilon)m$ edges.*

Proof To prove the lemma it is enough to show that if the transitive closure of G' has more than $12m - \epsilon m$ edges, we can extract from G' an assignment f for ϕ which satisfies at least a $1 - O(\epsilon)$ fraction of clauses. For the rest of the proof we assume that the transitive closure of G' has more than $12m - \epsilon m$ edges.

First, observe that each connected component of G' has size at most 4, since there are only 4 colors of the vertices in the graph. Also, notice that there are in total m vertices of color c, $6m$ vertices of color v (as the total number of literals in the formula ϕ equals $3m$), $3m$ vertices of color a, and $3m$ vertices of color b.
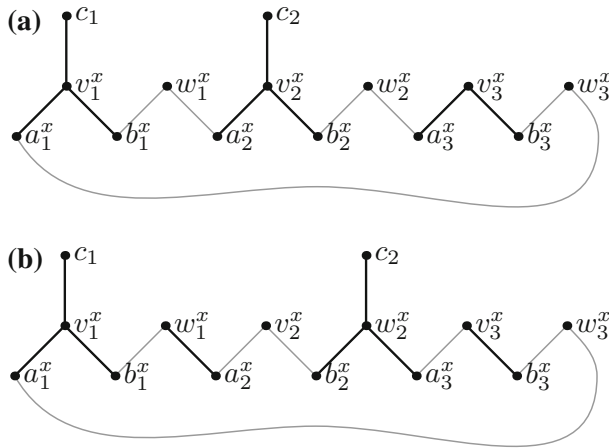
We now show that G' has at least $(1 - \epsilon)m$ connected components consisting of 4 vertices. Let $\alpha_1, \alpha_2, \alpha_3$ and α_4 denote the number of connected components of G' of size 1, 2, 3 and 4, respectively. The number of edges in the transitive closure of G' equals OPT $= 6\alpha_4 + 3\alpha_3 + \alpha_2$. As G' has $7m$ vertices of color other than v, exactly 3 such vertices are in each component of size 4, at least two such vertices are in each component of size 3, and at least one such vertex is in each component of size 2, we get: $\alpha_3 \leq (7m - 3\alpha_4)/2$ and $\alpha_2 \leq (7m - 3\alpha_4 - 2\alpha_3)$. We get

$$\text{OPT} = 6\alpha_4 + 3\alpha_3 + \alpha_2 \leq 7m + 3\alpha_4 + \alpha_3 \leq 10.5m + 1.5\alpha_4 \ .$$

As we assumed OPT $> 12m - \epsilon m$, we get that $\alpha_4 \geq (1 - \epsilon)m$.

Let us now consider a subgraph G_x' of G' corresponding to the variable x. G_x' consists of vertices a_i^x, b_i^x, v_i^x and w_i^x for $1 = 1, \ldots, n_x$, and additionally of the clause-vertices c_j which are incident in G' with any of the vertices v_i^x and w_i^x. As each clause-vertex has degree at most 1 in G' (as all the neighbors of a clause-vertex have the same color v), it belongs to at most one subgraph G_x'. Notice that each edge of G' is contained in some subgraph G_x', and therefore the edges of the transitive closure of G' are the union of the edges of the transitive closures of G_x'.

We say that G_x' is *inconsistent* if there are two vertices v_i^x and w_j^x in G_x' (where possibly $i = j$), such that both are incident with some clause-vertices $c_{i'}$ and $c_{j'}$ in G'. We now show that G' has at most ϵm inconsistent subgraphs G_x'. Let $\alpha^x \leq n_x$ be the number of clause-vertices c_i which belong to G_x'. Apart from these α^x vertices of color c, G_x' has also n_x vertices of colors a and b, and $2n_x$ vertices of color v. It is straightforward to verify that the transitive closure of G_x' has at most $3\alpha^x + 3n_x$ edges (see Fig. 4a for an example, where this bound is obtained). Moreover, if G_x' is inconsistent, its transitive closure has at most $3\alpha^x + 3n_x - 1$ edges (see Fig. 4b). If G' has

(a)

(b)

Fig. 4 **a** Graph G'_x maximizing the possible number of edges ($3\alpha^x + 3n_x$) in the transitive closure. **b** An inconsistent graph G'_x cannot achieve $3\alpha^x + 3n_x$ edges in the transitive closure

more than ϵm inconsistent subgraphs G'_x, we have OPT $\leq \sum_x 3\alpha^x + \sum_x 3n_x - \epsilon m = 12m - \epsilon m$, which contradicts our assumption.

We now extract from G' an assignment f for ϕ which satisfies at least a $(1 - \epsilon(\beta + 1))$ fraction of clauses. We proceed as follows. From our previous reasoning we know that G' has at least $(1 - \epsilon)m$ connected components consisting of 4 vertices, and that at most ϵm subgraphs G'_x are inconsistent. We fix the assignment f as follows. For each variable x, if G'_x is inconsistent or if G'_x does not contain any vertices c_i, we set $f(x)$ arbitrarily. Otherwise, either *all* vertices c_i from the component G'_x are incident with vertices v^x_j (corresponding to the literal x), or they are all incident with vertices w^x_j (corresponding to the literal $\neg x$). If the first case holds, we set $f(x)$ to TRUE. Otherwise, we set $f(x)$ to FALSE.

We now show a lower bound on the number of clauses satisfied by f. At least $(1 - \epsilon)m$ clause-vertices c_i are incident with some variable-vertex (as there are at least $(1 - \epsilon)m$ connected components of size 4). Each variable occurs at most $\beta = O(1)$ times in ϕ, and at most ϵm subgraphs G'_x are inconsistent, and therefore at most $\epsilon\beta m$ clause-vertices are incident with variables from an inconsistent subgraph. Therefore at least $m(1 - \epsilon(\beta + 1))$ clauses are satisfied by the assignment f.   □

Theorem 2 *The Maximum Edges in Transitive Closure (MEC) problem is APX-hard, even for graphs with only four colors.*

Proof Let ϕ be a MAX-3SAT(β) formula on m clauses, and $G = (V, E)$ a vertex-colored graph obtained from ϕ by our reduction. Let $G' = (V, E')$ be a subgraph of G which is an optimal solution for the MEC problem on G. From Lemma 4 we know, that if the formula ϕ is satisfiable, then the transitive closure of G' has at least $12m$ edges. From Lemma 5 we know, that if any assignment can satisfy at most a $(1 - \epsilon)$ fraction of the m clauses of ϕ, then the transitive closure of G' has at most $12m - \Theta(\epsilon)m$ edges.

As the $MAX - 3SAT(\beta)$ problem is APX-hard [1], and in particular for some constant $\epsilon > 0$ it is NP-hard to distinguish between the instances of $MAX - 3SAT(\beta)$ where all clauses are satisfiable, and the instances where at most a $(1 - \epsilon)$ fraction of the clauses can be satisfied, we obtain that MEC is APX-hard. □

4 Hardness of MCC

In this section we prove that the MCC problem does not admit polynomial-time approximation within a factor of $|V|^{1/14-\epsilon}$, for any $\epsilon > 0$, unless $P = NP$, or within a factor of $|V|^{1/2-\epsilon}$, unless $ZPP = NP$. The results hold even if each vertex color appears at most two times in the input graph. We prove our results via a reduction from the Minimum Clique Partition problem.

Minimum Clique Partition: Given a simple, undirected graph $G = (V, E)$, find a partition of V into a minimum number of subsets $V_1, \ldots, V_k$ such that the subgraph of G induced by each set of vertices V_i is a complete graph.

The Minimum Clique Partition problem is equivalent to Minimum Graph Coloring [9], and therefore it cannot be approximated in polynomial time within a factor of $|V|^{1/7-\epsilon}$ for any $\epsilon > 0$ [3], unless $P = NP$, or within a factor of $|V|^{1-\epsilon}$, unless $ZPP = NP$ [6].

4.1 Reduction from Minimum Clique Partition

Let $G = (V, E)$ be an instance of the Minimum Clique Partition problem. We create an instance of the MCC problem, i.e., a vertex colored graph $G' = (V', E')$, as follows. The reduction is illustrated in Fig. 5.

1. The vertex set $V' = V_b' \cup V_a'$ consists of two parts. The set $V_b' = V$ is the set of all vertices in G, each colored with a distinct color. We term these vertices *base vertices*. The set V_a' has two vertices, u_v and v_u, for each pair of vertices $u, v \in V$ such that $(u, v) \notin E$. Both vertices u_v and v_u have the same color, which is different from other colors in the graph. We refer to the vertices from V_a' as *additional vertices*. We emphasize that each color appears *at most* two times in G'.
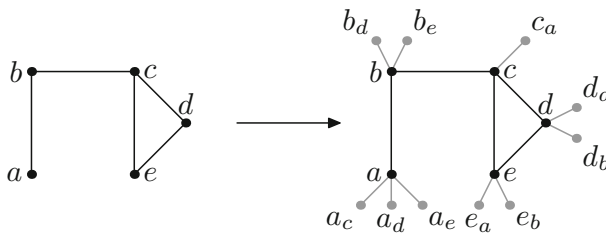
Fig. 5 Creating an instance of the MCC problem (*right*) from an instance of the Minimum Clique Partition (*left*). Base vertices and edges are drawn in *black*, and the additional ones in *gray*. An optimal solution for both problems is obtained by removing an edge (b, c)

2. The set of edges $E' = E'_b \cup E'_a$ consists of two parts. First, $E'_b = E$ is the set of edges in G, which we term *base edges*. The set E'_a has two edges, (u_v, u) and (v_u, v), for each pair of vertices $u, v \in V$ such that $(u, v) \notin E$ (i.e., each additional vertex u_v is connected with a base vertex u). We refer to the edges from E'_a as *additional edges*.

4.2 Analysis of the Reduction

Let $G = (V, E)$ be an instance of the Minimum Clique Partition problem, and $G' = (V', E')$ the corresponding instance of MCC, obtained by our reduction. We first compare the costs of the optimal solution for both problem instances, which leads to the main theorem of this section.

Lemma 6 *If there is a partition of G into k cliques, then the optimal solution for the MCC problem for G' has cost at most k.*

Proof Let G be a graph which can be partitioned into k cliques. We have to show that there is a collection of edges $E'' \subseteq E'$ in G', such that after removing E'' from G' we obtain a graph consisting of at most k colorful components. The set of edges E'' is exactly the set of base edges that have been removed from G to obtain the collection of k cliques.

As we do not remove any additional edges of G' (i.e., the edges from the set V'_a), the resulting graph consists of k connected components. The only pairs of vertices sharing the same color are pairs u_v, v_u such that $u, v \in V$ and $(u, v) \notin E$. Then u and v must be in different connected components of the clique partition, and so u and v (and therefore also u_v and v_u) are in different connected components of the constructed graph. Each connected component of the constructed graph is colorful. □

Lemma 7 *If the optimal solution for the MCC problem for G' has cost k, then there exists a partition of G into k cliques.*

Proof Let G' be a graph which can be transformed, by removing a collection of edges $E'' \subseteq E'$, into a graph consisting of k connected colorful components. We show that we can modify E'', without increasing the number of connected components in the resulting graph and while ensuring that each connected component stays colorful, so that E'' does not contain any edge from the set of additional edges E'_a. Then, by removing E'' from G, we obtain a valid partition of G into at most k cliques: For any pair of vertices $u, v \in V$ such that $(u, v) \notin E$, there are two vertices u_v and v_u in G', sharing the same color and connected with u and v, respectively, via additional edges. As no additional edges are contained in E'' and each connected component of $(V', E' \setminus E'')$ is colorful, u and v must be in different connected components of $(V', E' \setminus E'')$. In the partition of G the vertices u and v are then also in different connected components, and so we obtain a partitioning of G into at most k cliques.

We now show how to modify E''. For each additional edge $e = (u, u_v) \in E'_a$ which is in E'' we perform the following operation. First, we remove e from E''. That decreases the number of connected components by one, but might result in an infeasible solution. However, the only pair of vertices of the same color which are in

the same connected component of $(V', E' \setminus E'')$ can now be u_v and v_u. Denote by C the connected component of $(V', E' \setminus E'')$ containing u_v and v_u, and therefore also u and v. We now find a minimum cut separating u from v in C, and add the edges of the cut to E''. That results in splitting C into exactly two connected components, and each of the two components is colorful.

We perform the above operation for each additional edge $e = (u, u_v) \in E''$, and at the end we obtain a set E'' satisfying the needed conditions. By removing E'' from G we get a partition of G into at most k cliques. □

Theorem 3 *The minimum colorful components (MCC) problem does not admit polynomial-time approximation within a factor of $n^{1/14-\epsilon}$, for any $\epsilon > 0$, unless $P = NP$, or within a factor of $n^{1/2-\epsilon}$, for any $\epsilon > 0$, unless $ZPP = NP$, where n is the number of vertices in the input graph.*

Proof Let $G = (V, E)$ be an instance of the Minimum Clique Partition problem, and $G' = (V', E')$ the corresponding instance of the MCC problem, obtained by our reduction. From Lemmas 6 and 7 we obtain that the cost of an optimal solution for the MCC problem for G' is the same as the cost of an optimal solution for the Minimum Clique Partition problem for G. We know that the Minimum Clique Partition problem is hard to approximate within a factor of $|V|^{1/7-\epsilon}$, unless $P = NP$, or within a factor of $|V|^{1-\epsilon}$, unless $ZPP = NP$, were $|V|$ is the number of vertices in the graph G. Since G' has $|V'| \leq |V|^2$ vertices, our theorem follows. □

5 Open Problems

The APX-hardness result for the MEC problem requires that the input graphs are colored with at least four colors. A natural question is, thus, to settle the complexity of the problem for three colors (as for the case of two colors MEC is easily solvable in polynomial time, using a maximum matching algorithm). Another open question is to design approximation algorithms for the MEC problem or to strengthen the hardness of approximation result.

From the biological perspective it is interesting to analyze how our MSV algorithm behaves on real data. Finally, we mention that an intriguing and challenging task is to find problems in the colorful components framework that admit practical algorithms and are also meaningful for the biological applications.

References

1. Ausiello, G., Protasi, M., Marchetti-Spaccamela, A., Gambosi, G., Crescenzi, P., Kann, V.: Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties, 1st ed. Springer, New York (1999)
2. Avidor, A., Langberg, M.: The multi-multiway cut problem. Theoretical Computer Science **377**(1–3), 35–42 (2007)
3. Bellare, M., Goldreich, O., Sudan, M.: Free bits, PCPs, and nonapproximability—towards tight results. SIAM J. Comput. **27**(3), 804–915 (1998)

4. Bruckner, S., Hüffner, F., Komusiewicz, C., Niedermeier, R.: Evaluation of ILP-based approaches for partitioning into colorful components. In: Bonifaci, V., Demetrescu, C., Marchetti-Spaccamela, A. (eds.) SEA, volume 7933 of Lecture Notes in Computer Science, pp. 176–187. Springer, Berlin (2013)
5. Bruckner, S., Hüffner, F., Komusiewicz, C., Niedermeier, R., Thiel, S., Uhlmann, J.: Partitioning into colorful components by minimum edge deletions. In: Kärkkäinen, J., Stoye, J., (eds.) CPM, volume 7354 of Lecture Notes in Computer Science, pp. 56–69. Springer, Berlin (2012)
6. Feige, U., Kilian, J.: Zero knowledge and the chromatic number. J. Comput. Syst. Sci. **57**(2), 187–199 (1998)
7. He, G., Liu, J., Zhao, C.: Approximation algorithms for some graph partitioning problems. J. Graph Algorithms Appl. **4**(2), 1–11 (2000)
8. Mushegian, A.R.: Foundations of Comparative Genomics. Elsevier, Amsterdam (2010)
9. Paz, A., Moran, S.: Non deterministic polynomial optimization problems and their approximations. Theor. Comput. Sci. **15**(3), 251–277 (1981)
10. Sankoff, D.: OMG! Orthologs for multiple genomes—competing formulations—(keynote talk). In: Chen, J., Wang, J., Zelikovsky, A. (eds.) ISBRA, volume 6674 of Lecture Notes in Computer Science, pp. 2–3. Springer, Berlin (2011)
11. Savard, O.T., Swenson, K.M.: A graph-theoretic approach for inparalog detection. BMC Bioinform. **13**(S-19), S16 (2012)
12. Zheng, C., Swenson, K.M., Lyons, E., Sankoff, D.: OMG! Orthologs in multiple genomes—competing graph-theoretical formulations. In: Przytycka, T.M., Sagot, M.-F. (eds.) WABI, volume 6833 of Lecture Notes in Computer Science, pp. 364–375. Springer, Berlin (2011)