Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 108C (2017) 1115–1124

Procedia Computer Science

International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland

# Orthology Correction for Gene Tree Reconstruction: Theoretical and Experimental Results

Riccardo Dondi[1], Giancarlo Mauri[2], and Italo Zoppis[2]

[1] Università degli Studi di Bergamo, Bergamo, Italy
riccardo.dondi@unibg.it
[2] Università degli Studi di Milano-Bicocca, Milano, Italy
mauri@disco.unimib.it, zoppis@disco.unimib.it

**Abstract**

We consider how the orthology/paralogy information can be corrected in order to represent a gene tree, a problem that has recently gained interest in phylogenomics. Interestingly, the problem is related to the Minimum CoGraph Editing problem on the relation graph that represents orthology/paralogy information, where we want to minimize the number of edit operations on the given relation graph in order to obtain a cograph. In this paper we provide both theoretical and experimental results on the Minimum CoGraph Editing problem. On the theoretical side, we provide approximation algorithms for bounded degree relation graphs, for the general problem and for the problem restricted to deletion of edges. On the experimental side, we present a genetic algorithm for Minimum CoGraph Editing and we provide an experimental evaluation of the genetic algorithm on synthetic data.

© 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the International Conference on Computational Science

*Keywords:* Orthologous Genes, Gene Tree Reconstruction, CoGraph Editing, Approximation Algorithms, Genetic Algorithms

## 1 Introduction

Understanding the relations among homologous genes is fundamental in different contexts, for example to predict gene functionality. When studying gene functionalities, a family of *homologs* is usually considered, where two genes are called *homologs* if they originated from the same ancestral gene. However, different evolutionary events may have given rise to two homologous genes: losses, duplications, speciations, lateral gene transfers. In particular, if the ancestral gene of two homologous genes is a duplication (speciation, respectively), the two genes are called *paralogous* (*orthologous*, respectively). The evolution of a set of homologous genes is represented with a phylogenetic tree, called *gene tree*, whose leaves are labeled by the set of genes and whose internal nodes represent the evolutionary events (speciations, duplications). The comparison of a gene tree and of a species tree (the phylogenetic tree that represents the evolutionary history

1877-0509 © 2017 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of the scientific committee of the International Conference on Computational Science
10.1016/j.procs.2017.05.047

of the species considered), allows to infer the evolutionary events responsible for the evolution of the set of genes studied.

A popular conjecture, called orthologs conjecture [20], states that orthologous genes, are usually more similar both in the sequence and in functionality than paralogous genes. This motivates the research of reliable methods to infer the kind of relations among homologous genes.

Two approaches have been considered for the inference of the relations among homologous genes: (1) reconciliation-based methods, and (2) clustering methods. The first approach, see [8, 3], is based on the comparison between a gene tree for the considered family and a species tree. Clustering methods are instead based on sequence similarity (see for example [22, 15, 2, 14]). However, both methods are known to be error-prone (see [9, 11, 4, 18]), thus a major problem in phylogenomics is the integration of the two approaches, in order to infer reliable sets of gene relations. Some methods have investigated the integration of these two approaches [9, 10, 13, 12], giving interesting results based on graph-theory. Giving a graph $R_G$ (called *relation graph*) representing a set $R$ of relations among genes obtained via clustering methods, a first problem defined in this direction asks whether a set of relations among genes is representable with a (gene) tree. A set of relations that admits a representation with a gene tree is called *satisfiable*. Interestingly, in [9, 13] it is proved that a set $R$ of relations is satisfiable if and only if the graph $G_R$ that represents $R$ is a *cograph*, that is a graph that does not contain simple paths of length four.

Since the clustering methods to detect gene relations are error-prone, we consider the problem of correcting a set of orthologous/paralogous relations. The corresponding combinatorial problem is the Minimum CoGraphEditing problem, that, giving a graph, asks for the minimum number of edge insertion and edge deletion so that the resulting graph is a cograph. Minimum CoGraphEditing is NP-hard [16], and it is approximable within factor $4\Delta$, where $\Delta$ is the degree of the input graph [19], and in factor $n$ [5], where $n$ is the number of vertices of the graph. In [16], the parameterized complexity of the problem has been considered, showing that Minimum CoGraphEditing is fixed-parameter tractable when parameterized by the number of edit operations. Moreover, an integer linear programming has been considered in [10]. Notice that recently some results on the approximation of the problem have been given for its weighted variant [5].

Here, we give both theoretical and practical results for Minimum CoGraphEditing. On the theoretical side, we present in Section 3.2 an approximation algorithm of factor $3\Delta$, where $\Delta$ is the degree of the input graph, for the problem and an approximation algorithm of factor $2\Delta$ for the deletion variant of the problem (that is the variant where edges can be only removed and no edge insertion is allowed). Notice that the deletion variant of the problem is NP-hard [6]. The two algorithms are based on the isolation of vertices of the graph and have a factor depending on $\Delta$, thus their application in practice is limited. Hence, we consider in Section 4 genetic algorithms as an approach to compute efficiently optimal or near optimal solution for the Minimum CoGraphEditing problem. We give in Section 4 a natural genetic algorithm, and we present some experimental results on simulated data. The results show that usually the genetic algorithm is efficient, but the quality of returned solutions is influenced by the skewness in the degree distribution of the input graph. We conclude the paper with some open problems. We start by introducing some notations and by formally defining the problem in Section 2.

## 2 Definitions

In the section we introduce the main concepts related to orthology, and we define formally the problem we are interested in. We consider trees that are rooted and that, except for the root, have no nodes of degree 2. The leafset of a tree $T$ is denoted by $L(T)$. Given a set of elements $L$, a tree *labeled* by $L$ is a tree $T$ such that there exists a bijection between $L(T)$ and $L$. Given a set $\Gamma$ of homologous genes on a set $\Sigma$ of species, a *gene tree* $T_G$ represents the evolution of genes in $\Gamma$ and is a tree labeled by $\Gamma$. A species tree $T_S$ represents the evolution of the species in $\Sigma$ and is a tree labeled by $\Sigma$. The *lca*-mapping [7] allows to define the evolutionary events (here we consider only speciations and duplications) associated with the internal nodes of $T_G$, by mapping nodes of $T_G$ in nodes of $T_S$. Given a set of genes $\Gamma$ and a gene tree $T_G$ labeled by $\Gamma$, two genes $x, y \in \Gamma$ are *orthologous* if the lca of $x$ and $y$ in $T_G$ is a speciation; two genes $x$ and $y$ are *paralogous* if the lca of $x$ and $y$ in $T_G$ is a duplication.

Given a set of genes $\Gamma$, we represent their relations (orthologous/paralogous) with a graph, called *relation graph* and denoted as $R_G(\Gamma)$ or simply $R_G$ when it is clear from the context. $R_G = (V_G, E_G)$ where

$$V_G = \{v_x : \ x \text{ is in } \Gamma \ \}$$

$$E_G = \{\{v_x, v_y\} : v_x, v_y \in V_G, \ x \text{ and } y \text{ are orthologous } \}$$

Following the notation of [13], the ortholog relations is represented by an edge, while a paralog relation is represented by a missing edge.

Given a gene tree, it is always possible to reconstruct the corresponding relation graph. On the other side, given a relation graph, a corresponding gene tree may not exist. A relation graph for which there exists a corresponding gene tree is called *satisfiable*. In [9], it is shown that a relation graph $R_G$ is satisfiable if and only if it is a *cograph*, that is there is no set of exactly four vertices of $R_G$ that induces a simple path of length 4. A simple path of length 4 in a graph is called a $P_4$. Given four vertices $v_{i,1}, v_{i,2}, v_{i,3}, v_{i,4}$ of $R_G = (V_G, E_G)$ that induce a $P_4$, we denote by $\langle v_{i,1}, v_{i,2}, v_{i,3}, v_{i,4} \rangle$ the path connecting $v_{i,1}$ to $v_{i,2}$, $v_{i,2}$ to $v_{i,3}$ and $v_{i,3}$ to $v_{i,4}$, that is $\{v_{i,1}, v_{i,2}\}$, $\{v_{i,2}, v_{i,3}\}$, $\{v_{i,3}, v_{i,4}\} \in E_G$, while $\{v_{i,1}, v_{i,3}\}$, $\{v_{i,1}, v_{i,4}\}$, $\{v_{i,2}, v_{i,4}\} \notin E_G$.

Given a graph $G_R = (V_G, E_G)$, an *edge insertion* consists of adding an edge $\{v_i, v_j\}$, with $v_i, v_j \in V_G$, where $\{v_i, v_j\} \notin E_G$; an *edge deletion* consists of removing an edge $\{v_i, v_j\} \in E_G$. A *relation correction* of $R_G$ is either an edge insertion or an edge deletion.

Now, we are able to introduce the problem we are interested in. The problem asks for the minimum number of relation corrections such that the resulting graph is a CoGraph.

**Minimum CoGraph Editing:**(MinCoED)

**Input:** A gene family $\Gamma$ and relation graph $R_G = (V_G, E_G)$ for $\Gamma$;
**Output:** A cograph $R'_G = (V'_G, E'_G)$ such that $V_G = V'_G$ and $|E_G \setminus E'_G + E'_G \setminus E_G|$ is minimum.

In the following, we consider also a variant of MinCoED, called **Minimum CoGraph by Deletion** (MinCoDEL), in which only edge deletions are allowed.

# 3 Approximation Algorithms for MinCoED and MinCoDEL

In this section, we present approximation algorithms for MinCoED and MinCoDEL. The approximation algorithms are inspired by that in [19] that achieves an approximation factor $4\Delta$ for MinCoED and MinCoDEL. The approximation algorithm in [19] is designed for a general variant of MinCoED, where the input graph has to be edited so that it does not contain a forbidden subgraph of size $t$, for some constant $t \geq 1$. The approximation algorithm in [19] greedily finds one occurrence of such a forbidden subgraph of size $t$ and it deletes all the edges incident in its vertices. In the case of MinCoED and MinCoDEL, the algorithm picks a $P_4$, and it deletes all the edges incident in the vertices of the $P_4$. This leads to an approximation of factor $4\Delta$. Here we show that for MinCoED and MinCoDEL, we can reduce the number of vertices we select for editing or deletion.

## 3.1 A $3\Delta$-Approximation Algorithm for MinCoED

In this section, we describe a $3\Delta$-approximation algorithm for MinCoED, where $\Delta$ is the degree of the input graph $G$. The approximation algorithm, called APPR-ALG-ED, considers the graph $R_G$ and modifies it until it is a cograph, that is there is no $P_4$. Consider a $P_4$ in $R_G$, denoted by $p_i$, with $p_i = \langle v_{i,1}, v_{i,2}, v_{i,3}, v_{i,4} \rangle$; the set $\{v_{i,1}, v_{i,2}, v_{i,3}\}$ is called the *pivot triplet* of $p_i$ and is denoted by $t(p_i)$.

While $R_G$ is not a cograph, the algorithm greedily picks a $P_4$ $p_i$ in $R_G$, and deletes all the edges incident in the vertices of $t(p_i)$. The iteration is repeated until the resulting graph does not contain any $P_4$. The time complexity of the algorithm is clearly polynomial, as it is dominated by the time required to compute if there exists a $P_4$ in $R_G$, which requires at most time $O(n^4)$.

Now, we show that APPR-ALG-ED provides an approximation algorithm of factor $3\Delta$. First, we prove a lower bound on the number of relation corrections required to obtain a cograph starting from $G$.

**Lemma 1.** *Given a set $\mathcal{P}$ of $P_4$s in $R_G$, such that, for each $p_i, p_j \in \mathcal{P}$, $p_i$ and $p_j$ share at most one vertex. Then, a cograph $R'_G$ is obtained from $R_G$ with at least $|\mathcal{P}|$ relation corrections.*

*Proof.* Consider $p_i$, $p_j$ in $\mathcal{P}$. Notice that, since $p_i$ and $p_j$ share at most one vertex, a relation modification in $p_i$ affects two vertices $\{v_{i,x}, v_{i,y}\}$ of $p_i$, and at most one of $v_{i,x}, v_{i,y}$ belongs to $p_j$. Hence, the modification of $\{v_{i,x}, v_{i,y}\}$ does not affect any relations in $p_j$. □

Now, we focus on the set $\mathcal{T}$ of $P_4$s selected by APPR-ALG-ED. We prove the following property.

**Lemma 2.** *Let $\mathcal{T}$ be the set of $P_4$s selected by APPR-ALG-ED. Then for each $p_i, p_j \in \mathcal{T}$, $p_i$ and $p_j$ share at most one vertex.*

*Proof.* Consider two $P_4$s $p_i$, $p_j$ in $\mathcal{T}$. Assume without loss of generality that $p_i$ was selected before $p_j$. Let $p_i = \langle v_{i,1}, v_{i,2}, v_{i,3}, v_{i,4} \rangle$. Since APPR-ALG-ED deletes all the edges incident in the triplet pivot $t(p_i)$, it follows that when $p_j$ is selected by the algorithm, the vertices in $t(p_i)$ are isolated. Hence, $p_j$ can contain at most vertex $v_{i,4}$. □

Now, we prove a lemma that shows that by deleting all the edges incident in a set of vertices of $R_G$, we do not induce any $P_4$ that was not in the original graph.

**Lemma 3.** *Let $W$ be a set of vertices in $R_G$ and let $R'_G$ the graph obtained from $R_G$ by deleting all the edges incident in $W$. Then every $P_4$ of $R'_G$ is also a $P_4$ in $R_G$.*

*Proof.* Let $p_i = \langle v_{i,1}, v_{i,2}, v_{i,3}, v_{i,4} \rangle$ a $P_4$ in $R'_G$ that is not a $P_4$ in $R_G$. Notice that, since by construction the set of edges of $R'_G$ is a subset of the set of edges of $R_G$ and since $p_i$is not a $P_4$ in $R_G$, there exists an edge connecting two vertices, w.l.o.g. $v_{i,1}$, $v_{i,3}$, in $R_G$ that is not an edge of $R'_G$. It follows that one of $v_{i,1}$, $v_{i,3}$ belongs to $W$, w.l.o.g. $v_{i,1}$. Thus by construction $v_{i,1}$ is an isolated vertices of $R'_G$ and cannot belong to a $P_4$.

□

Now, we denote by $OPT$ the value of an optimal solution of MinCoED over instance $R_G$, and we show that APPR-ALG-ED returns a solution having approximation factor $3\Delta$.

**Theorem 1.** *APPR-ALG-ED returns a solution of MinCoED that modifies at most $3\Delta \cdot OPT$ relations.*

*Proof.* By Lemma 2 and Lemma 3, the set $\mathcal{T}$ of $P_4$s selected by APPR-ALG-ED consists of $P_4$s contained in $R_G$ such that, for each $p_i, p_j \in \mathcal{T}$, they share at most one vertex. By Lemma 1 $OPT \geq |\mathcal{T}|$ and, since APPR-ALG-ED deletes at most $3\Delta$ edges for each $p_i \in \mathcal{T}$, the theorem holds.

□

## 3.2 A $2\Delta$-Approximation Algorithm for MinCoDEL

Now, we consider the MinCoDEL problem and we show that the approximation factor can be further reduced to $2\Delta$.

First, we describe the approximation algorithm, denoted by APPR-ALG-DEL, which is very similar to that of the previous section. Similarly, APPR-ALG-DEL considers the graph $R_G$ and modifies it until it is a cograph, that is there is no $P_4$ in the resulting graph. First consider $p_i$, a $P_4$ in $R_G$, with $p_i = \langle v_{i,1}, v_{i,2}, v_{i,3}, v_{i,4} \rangle$; the set of vertices $v_{i,2}$, $v_{i,3}$ is called the *pivot pair* of $p_i$ and is denoted by $pp(p_i)$.

APPR-ALG-DEL greedily picks a $P_4$ $p_i$ in $R_G$ and deletes all the edges incident in vertices of $pp(p_i)$. The iteration is repeated until the resulting graph does not contain any $P_4$. As in the previous algorithm, the time complexity of the algorithm is clearly polynomial.

Now, we show that APPR-ALG-DEL provides an approximated solution of factor $2\Delta$. First, we prove a lower bound on the number of edit operations required to obtain a cograph starting from $G$.

**Lemma 4.** *Given a set $\mathcal{P}$ of edge disjoint $P_4$s in $R_G$, then a cograph $R'_G$ is obtained from $R_G$ with at least $|\mathcal{P}|$ edge deletions.*

*Proof.* Consider $p_i, p_j \in \mathcal{P}$. Since $p_i$ and $p_j$ are edge disjoint, it follows that the deletion of an edge of $p_i$ does not delete any edge of $p_j$. As a consequence, $p_j$, after the deletion of some edge of $p_i$, is still a $P_4$. It follows that at least $|\mathcal{P}|$ edge deletions are needed so that each $p_i \in \mathcal{P}$ is removed from $R_G$. □

Now, we prove that the set $\mathcal{W}$ of $P_4$s selected by the APPR-ALG-DEL are edge disjoint.

**Lemma 5.** *Let $\mathcal{W}$ be the set of $P_4$s in $R_G$ selected by APPR-ALG-DEL. Then $\mathcal{W}$ is a set of edge disjoint $P_4$s.*

*Proof.* Consider $p_i$, $p_j$ in $\mathcal{W}$, and assume without loss of generality that $p_i$ was picked by APPR-ALG-DEL before $p_j$. Since APPR-ALG-DEL deletes all the edges incident in the pair pivots of $p_i$, it follows that when $p_j$ is picked every edge of $p_i$ has been deleted. Thus $p_i$ and $p_j$ cannot share an edge. □

Now, denote by $OPT$ the value an optimal solution of MinCoDEL over instance $R_G$.

**Theorem 2.** *APPR-ALG-DEL returns a solution of MinCoDEL that deletes at most* $2\Delta \cdot OPT$ *relations.*

*Proof.* By Lemma 5 and Lemma 3, the set $\mathcal{W}$ consists of edge disjoint $P_4$s contained in $R_G$. By Lemma 4 $OPT \geq |\mathcal{W}|$ and, since APPR-ALG-DEL deletes at most $2\Delta$ edges for each $p_i \in \mathcal{W}$, the theorem holds. □

Notice that the analysis of the approximation factor of APPR-ALG-DEL cannot be directly extended to MinCoED, as Lemma 4 does not hold in this case. Indeed, in some cases it is possible to edit a set edge disjoint $P_4$s with a single edge insertion.

# 4 A Genetic Algorithm for MinCoED

The algorithms presented in Section 3 are mainly of theoretical interests. Indeed, although they represent simple and (probably) fast heuristics, they tend to isolate vertices of the graph, which is not a desirable property of a solution of MinCoED. On the other side, their approximation factors depends on $\Delta$, and for large graphs this may lead to solution far from the optimum. To this concern, we consider here a different approach based on genetic algorithms (GA) [17].

Generally, genetic algorithms use chromosomes in order to represent instances and compute solutions of the problem at hand. In our case, we coded in the chromosome population the instances of the MinCoED problem. The algorithm was coded in R using the Genetic Algorithm package (GA package) [21] [1]. More specifically the following issues were considered in our simulations.

- **Chromosome representation**

  In order to benefit most from the potentialities offered by the GA package, we adopted a very simple representation for chromosomes. In our script, chromosomes represent graphs through binary vectors which are obtained by combining rows of the graph adjacency matrices.

- **Fitness**

  A fitness function $f$ was defined to promote new chromosome generations with both the minimum number of edit operations to the input graphs (i.e. chromosome representation), and the minimum number of $P_4$s generated within the graphs represented through new chromosomal offspring. With this aim, we defined $f(NoP_4, y) = \exp -(k_1 * NoP_4 + k_2 * y)$, where $NoP_4$ represents the number of $P_4s$ in the graph associated with the chromosome, $k_1$ and $k_2$ are constants used to balance the contributes of the following two main components: $NoP_4$ and the difference $d = XOR(R_G, R'_G)/2$ between the input graph $R_G$ and the graph $R'_G$ obtained through the final best solution from the offspring representation.

[1] GA package is downloadable at `https://cran.r-project.org/web/packages/GA/index.html`

- **Mutation**

  Mutation process was generated with the following two objectives.

  - To introduce the minimum number of changes in the initial graph. This was simply provided by shifting bits from one to zero (respectively, from zero to one) over the coded chromosomes.
  - To apply the methodology discussed in Section 3.1 for the approximation of Min-CoED (with low probability). In this case, we greedily sampled $P4$s from represented graphs (i.e. using random walks over the graphs) and, for each sample, we deleted the edges incident to the pivot triplet of each selected $P_4$.

- **Crossover**

  The crossover operator simply moves a relation between two vertices from a graph to another. In other terms, given two adjacency matrices $M_1$ and $M_2$ (respectively associated with the graphs represented through chromosome 1 and 2) and $s = M_1[x, y]$, we have (after the crossover application) $M_2[x, y] = s$; the same applies to $M_2$, i.e if $s = M_1[x, y]$ then we have $M_2[x, y] = s$.

- Selection

  To guarantee that the solution quality does not decrease from one generation to another [1], we allowed best fitness individuals to survive at each generation (i.e. elitism).

## 4.1 Experimental Results

Genetic algorithms were applied to simulated data (graphs) using the ape package [2] and the igraph package [3]. We generated input graphs, first by randomly sampling gene trees, then transforming the gene trees into the corresponding relation graphs (which do not contain $P_4$s as proved in [9]). Finally, we added $P_4$s by perturbing the obtained graphs with a random mechanisms able to create (or delete) edges between vertices. Please notice that in this phase the number of added $P_4$s are strictly dependent to the graph structures. Targets of our simulations were the following:

- To asses whether the genetic algorithm results in a decrease in the added number of $P_4$s.
- To asses whether the genetic algorithm task is influenced in some way by structural properties of the input graphs.

In the first case, we applied GA to graphs with a number of vertices ranging in $S = \{15, 25, 35\}$, and for each $s \in S$ we repeated 10 tests with different input to get a better evaluation of the algorithms behavior. Results are reported in Table 1.

We can observe the following facts.

- Performances definitely benefit when the number of vertices of the input graph is lower than 35.
- The greater the number of paths introduced, the greater the difficulty of removing them

[2] https://cran.r-project.org/web/packages/ape

[3] http://igraph.org/r

| N. of Vertices | N. of $P_4$ - Start | N. of $P_4$ - Final | Distance | User time | System time |
|---|---|---|---|---|---|
| 15 | 54.20 | 0.80 | 7.9 | 86.74 | 0.31 |
| 25 | 904.60 | 12.50 | 46.60 | 195.99 | 0.82 |
| 35 | 3261.40 | 2528.40 | 147.80 | 200.02 | 0.99 |

Table 1: Summary of performances (average values); Distance: distance between the starting and the final graph in terms of edge insertion/removal. User time: seconds spent for calculations. System time: seconds spent by the OS to respond program's requests

.

- Both the distance and the computation time do not seem to be too heavy and do not change substantially with the vertex number.

It is reasonable to assume that GA difficulty in decreasing the number of paths is due to some structural property of the input graphs. Thus, in order to answer the second issue, we considered the degree distributions of the input. Indeed, we defined two new variables. The first for the skewness in the degree distribution: a variable called "skewnees class", taking value 1 for positive skewed distribution (0, otherwise). The second variable represents a GA negative behavior: a variable called "Risk Class", equal to 1 for negative performance. In these experiments we considered as negative, results for which GA was not able to reduce the number of $P_4$ at least of 50% compared to the initial number (respectively, positive results was coded by 0, if this was not the case). Please notice that, a negative skewed distribution of the vertex degree express higher connectivity in the input graphs (i.e. many nodes with a high degree of connectivity).
Experiments were conducted using 45 and 55 vertices for the input graphs. In this case we repeated 20 tests for 45 vertices and 10 tests for 55 vertices.

Figure 1 reports a $2 \times 2$ contingency table where data consist of two binary variables, one for the outcome (Risk Class) and one for the skewness (being used as predictor variable). The figure shows that while for negative skewed distribution the number of times in which GA has a negative behavior is predominant, many tests do not fall into the risk class 1 (i.e. good performances) for positive skewness (i.e. many nodes with low degree of connectivity). To give a significance evidence for these data we conducted a chi-square test (Pearson chi-square). The results are reported in Figure 2: a significant difference between the risk of falling into the level 1 for the Risk class variable and negative skewed distribution exists in patterns collected from the experiments. The p-value equals to 0.021, and this is smaller than the significance ($\alpha$) level 0.05. In Figure 3 we report summary statistics (average values) concerning the distance of the final solution from the input graphs, as well as the computational time. Also in this case the average values do not seem to be too burdensome.

# 5 Conclusion

We have considered two combinatorial variants (MinCoED and MinCoDEL) of the problem of editing a relation graph representing orthologous/paralogous relations among genes. We have provided both theoretical results (approximation algorithms for bounded degree graphs) and experimental results (a genetic algorithm). There are interesting open problems in both perspective. From a theoretical point of view, it would be interesting to consider if the approximation factor of MinCoED and MinCoDEL can be improved. On the experimental side, it would be interesting to test our genetic algorithm on real data and to design efficient heuristics

| Contingency table Skewness_Class * Risk_Class | | | | |
|---|---|---|---|---|
| Count | | | | |
| | | Risk class | | |
| | | 0 | 1 | Total |
| Skewness Class | 0 | 4 | 13 | 17 |
| | 1 | 9 | 4 | 13 |
| Total | | 13 | 17 | 30 |

Figure 1: $2 \times 2$ contingency table, where the values in the entries represent the number of tests corresponding to the outcome (Risk Class) and the skewness values.

| Chi-square test | | | | | |
|---|---|---|---|---|---|
| | Value | df | Asymp. Sig. (2-sided) | Exact Sig. (2-sided) | Exact Sig. (1-sided) |
| Pearson Chi-Square | 6,266[a] | 1 | ,012 | | |
| Continuity Correction | 4,543 | 1 | ,033 | | |
| Likelihood Ratio | 6,455 | 1 | ,011 | | |
| Fisher's Exact Test | | | | ,025 | ,016 |
| Linear-by-Linear Association | 6,057 | 1 | ,014 | | |
| | 30 | | | | |

Figure 2: The chi-square test (Pearson chi-square) give a p-value equals to 0.021 (column Asy. mp. sig. (2-sided)).

for the MinCoED problem.

# References

[1] S. Baluja and R. Caruana. Removing the genetics from the standard genetic algorithm. In A. Prieditis and S. J. Russell, editors, *Machine Learning, Proceedings of the Twelfth International*

| Descriptive Stat. | | |
|---|---|---|
| | SkewnessClass | Statistics |
| distance | 0 | 482,59 |
| | 1 | 336,23 |
| time.1. | 0 | 160,7624 |
| | 1 | 152,2792 |
| time 2. | 0 | ,2306 |
| | 1 | ,2938 |
| time 3. | 0 | 161,2600 |
| | 1 | 152,8162 |
| skewness. | 0 | -397172209,9672450000000 |
| | 1 | 101423471,0629568000000 |

Figure 3: Average values for distance, time and skewnees for both risk class 0 and 1; (Time 1 (User time): seconds spent for calculations. Time 2 (System time): seconds spent by the OS to respond program's requests. Time 3 (elapsed time): sum of User and System time, plus other program and/or the OS tasks.

*Conference on Machine Learning, Tahoe City, California, USA, July 9-12, 1995*, pages 38–46. Morgan Kaufmann, 1995.

[2] A.C. Berglund, E. Sjolund, G. Ostlund, and E.L. Sonnhammer. InParanoid 6: eukaryotic ortholog clusters with inparalogs. *Nucl. Acids Res.*, 36, 2008.

[3] G. Blin, P. Bonizzoni, R. Dondi, R. Rizzi, and F. Sikora. Complexity insights of the minimum duplication problem. *Theor. Comput. Sci.*, 530:66–79, 2014.

[4] R. Dondi, N. El-Mabrouk, and K. M. Swenson. Gene tree correction for reconciliation and species tree inference: Complexity and algorithms. *J. Discrete Algorithms*, 25:51–65, 2014.

[5] R. Dondi, M. Lafond, and N. El-Mabrouk. Approximating the correction of weighted and unweighted orthology and paralogy relations. *Algorithms for Molecular Biology*, 12(1):4, 2017.

[6] E. S. El-Mallah and C. J. Colbourn. The complexity of some edge deletion problems. *Circuits and Systems, IEEE Transactions on*, 35(3):354–362, 1988.

[7] W. M. Fitch. Homology. a personal view on some of the problems. *TIG*, 16(5):227- 231, 2000.

[8] M. Goodman, J. Czelusniak, G.W. Moore, A.E. Romero-Herrera, and G. Matsuda. Fitting the gene lineage into its species lineage, a parsimony strategy illustrated by cladograms constructed from globin sequences. *Syst. Zoology*, 28:132–163, 1979.

[9] M. Hellmuth, M. Hernandez-Rosales, K. Huber, V. Moulton, P. Stadler, and N. Wieseke. Orthology relations, symbolic ultrametrics, and cographs. *J. Math. Biol.*, 66(1–2):399–420, 2013.

[10] M. Hellmuth, N. Wieseke, M. Lechner, H. Lenhof, M. Middendorf, and P. F Stadler. Phylogenomics with paralogs. *PNAS*, 2014.

[11] M. Lafond, C. Chauve, R. Dondi, and N. El-Mabrouk. Polytomy refinement for the correction of dubious duplications in gene trees. *Bioinformatics*, 30(17):519–526, 2014.

[12] M. Lafond, R. Dondi, and N. El-Mabrouk. The link between orthology relations and gene trees: a correction perspective. *Algorithms for Molecular Biology*, 11:4, 2016.

[13] M. Lafond and N. El-Mabrouk. Orthology and paralogy constraints: satisfiability and consistency. *BMC Genomics*, 15(Suppl 6):S12, 2014.

[14] M. Lechner, S. Findeiß, L. Steiner, M. Marz, P.F. Stadler, and S.J. Prohaska. Proteinortho: detection of (co-)orthologs in large-scale analysis. *BMC Bioinformatics*, 12:124, 2011.

[15] L. Li, C.J. Jr. Stoeckert, and D.S. Roos. OrthoMCL: identification of ortholog groups for eukaryotic genomes. *Genome Research*, 13:2178- 2189, 2003.

[16] Y. Liu, J. Wang, J. Guo, and J. Chen. Complexity and parameterized algorithms for cograph editing. *Theor. Comput. Sci.*, 461:45–54, 2012.

[17] M. Mitchell. *An introduction to genetic algorithms*. Complex adaptive systems. MIT press, Cambridge (Mass.), 1996.

[18] A. Mykowiecka and P. Górecki. Bootstrapping algorithms for gene duplication and speciation events. In M. Botón-Fernández, C. Martín-Vide, S. Santander-Jiménez, and M. A. Vega-Rodríguez, editors, *Algorithms for Computational Biology - Third International Conference, AlCoB 2016, Trujillo, Spain, June 21-22, 2016, Proceedings*, volume 9702 of *Lecture Notes in Computer Science*, pages 106–118. Springer, 2016.

[19] A. Natanzon, R. Shamir, and R. Sharan. Complexity classification of some edge modification problems. *Discrete Applied Mathematics*, 113(1):109–128, 2001.

[20] S. Ohno. *Evolution by gene duplication*. Springer, Berlin, 1970.

[21] L. Scrucca. GA: A package for genetic algorithms in R. *Journal of Statistical Software*, 53(4):1–37, 2013.

[22] R.L. Tatusov, M.Y. Galperin, D.A. Natale, and E.V. Koonin. The COG database: a tool for genome-scale analysis of protein functions and evolution. *Nucl. Acids Res.*, 28:33- 36, 2000.