Rubert and Braga
Algorithms for Molecular Biology (2023) 18:14
https://doi.org/10.1186/s13015-023-00238-y

Algorithms for Molecular Biology

RESEARCH

Open Access

Efficient gene orthology inference via large-scale rearrangements

Diego P. Rubert[1,2] and Marília D. V. Braga[2*]

Abstract

Background Recently we developed a gene orthology inference tool based on genome rearrangements (*Journal of Bioinformatics and Computational Biology* 19:6, 2021). Given a set of genomes our method first computes all pairwise gene similarities. Then it runs pairwise ILP comparisons to compute optimal gene matchings, which minimize, by taking the similarities into account, the weighted rearrangement distance between the analyzed genomes (a problem that is NP-hard). The gene matchings are then integrated into gene families in the final step. The mentioned ILP includes an optimal *capping* that connects each end of a linear segment of one genome to an end of a linear segment in the other genome, producing an exponential increase of the search space.

Results In this work, we design and implement a heuristic capping algorithm that replaces the optimal capping by clustering (based on their gene content intersections) the linear segments into $m \geq 1$subsets, whose ends are capped independently. Furthermore, in each subset, instead of allowing all possible connections, we let only the ends of content-related segments be connected. Although there is no guarantee that m is much bigger than one, and with the possible side effect of resulting in sub-optimal instead of optimal gene matchings, the heuristic works very well in practice, from both the speed performance and the quality of computed solutions. Our experiments on primate and fruit fly genomes show two positive results. First, for complete assemblies of five primates the version with heuristic capping reports orthologies that are very similar to the orthologies computed by the version of our tool with optimal capping. Second, we were able to efficiently analyze fruit fly genomes with incomplete assemblies distributed in hundreds or even thousands of contigs, obtaining gene families that are very similar to FlyBase families. Indeed, our tool inferred a higher number of complete cliques, with a higher intersection with FlyBase, when compared to gene families computed by other inference tools. We added a post-processing for refining, with the aid of the MCL algorithm, our *ambiguous* families (those with more than one gene per genome), improving even more the accuracy of our results. Our approach is implemented into a pipeline incorporating the pre-computation of gene similarities and the post-processing refinement of ambiguous families with MCL. Both the original version with optimal capping and the new modified version with heuristic capping can be downloaded, together with their detailed documentations, at https://gitlab.ub.uni-bielefeld.de/gi/FFGC or as a Conda package at https://anaconda.org/bioconda/ffgc.

Keywords Comparative genomics, Double-cut-and-join, Indels, Gene orthology

*Correspondence:
Marília D. V. Braga
mbraga@cebitec.uni-bielefeld.de
[1] Faculdade de Computação, Universidade Federal de Mato Grosso do Sul, Campo Grande, Brazil
[2] Faculty of Technology and Center for Biotechnology (CeBiTec), Bielefeld University, Bielefeld, Germany

Background

The study of distances and parsimonious evolutionary scenarios based on large-scale genome rearrangements traditionally depends on the pre-computation of *gene families*. Computing such a distance is usually polynomial when genomes have at most one gene per family

[1–3] or NP-hard otherwise [4–8]. These works adopt several rearrangement models and among the most popular ones is the double-cut-and-join (DCJ) operation [9], which mimics organizational rearrangements, such as inversions, fusions, fissions and translocations.

In our research group an alternative (NP-hard) *family-free* setting for genome rearrangement approaches was proposed in 2013 [10]. Our studies were further extended, resulting in a model that does not require the pre-computation of gene families and, besides DCJ operations, takes into account insertions and deletions of DNA segments, collectively called *indels* [11, 12]. This model is able to infer pairwise orthologs between two genomes directly, simultaneously based on gene similarities and rearrangements. In practice, its optimization function can be solved exactly due to an ILP formulation [12] that is called FF-DCJ-Indel and also reports an optimal matching of orthologs between the two analyzed genomes. (The ILP FF-DCJ-Indel is itself based on the previous formulations for family-based approaches [7, 8].)

With these achievements we were able to invert the traditional paradigm of genome rearrangement studies: instead of requiring the gene families to proceed with rearrangement comparisons, it became possible to use rearrangement comparisons for inferring the gene families.[1] Indeed, in our previous work [13], we did a first attempt of using FF-DCJ-Indel for inferring genome-scale gene families across several species. More precisely, given a set of genomes, our method first computes all pairwise optimal gene matchings, which are integrated into gene families in the second step, resulting in a complete pipeline called OrthoFFGC,[2] whose inferences displayed good quality in the analysis of completely assembled genomes.

However, the integrated FF-DCJ-Indel may not converge in some cases, in particular when the number of segments of a genome is large, e.g. for genomes that are not completely assembled but split in several contigs. The main reason is that each ILP pairwise comparison includes an optimal *capping* that must allow the end of any linear segment of one genome to be matched to the end of any linear segment of the other genome. The optimal capping then produces an exponential increase of the search space.

In this work, we design and implement a heuristic capping algorithm that replaces the optimal capping by clustering (based on their gene content intersections) the linear segments into $m \geq 1$ subsets, so that the ends of the linear segments in the same subset S can only be matched to elements of S. Furthermore, in each subset, instead of allowing all possible connections, we let only the ends of content-related segments be connected. Although there is no guarantee that m is much bigger than one, and with the possible side effect of resulting in sub-optimal instead of optimal gene matchings, the heuristic works very well in practice, from both the speed performance and the quality of computed solutions.

We call OrthoFFGC≈ the new complete pipeline adopting the heuristic capping for FF-DCJ-Indel. Our experiments on five primate genomes show that for complete assemblies OrthoFFGC≈ reports orthologies that are very similar to those computed by OrthoFFGC, which is the version of our tool with optimal capping. Moreover, with OrthoFFGC≈ we can efficiently analyze fruit fly genomes with incomplete or heavily fragmented assemblies distributed in hundreds or even thousands of contigs. Considering 11 fruit fly genomes whose assemblies are split into 507 contigs on average and using FlyBase orthologies (https://flybase.org) as reference, we compared the gene families inferred by OrthoFFGC≈ to Oma [14, 15], ProteinOrtho [16], and Poff [17]. Our results show that OrthoFFGC≈ inferred a higher number of complete cliques of genes, with a higher intersection with FlyBase, when compared to gene families computed by other inference tools.

This paper is an extended version of a work recently presented at WABI [18]. In this extension, besides the already mentioned analysis of five primate genomes, we added a post-processing step that was not included in the work presented at WABI: refining, with the aid of the mcl algorithm [19], our *ambiguous* families (those with more than one gene per genome), which improved even more the accuracy of our results. Another relevant point of the extension presented here is the optimization of several aspects of our implementation, achieving running times that are similar to the fastest alternative tools ProteinOrtho and Poff for the *Drosophila* dataset, despite the use of pairwise ILP computations.

[1] Another attempt called msoar (Shi, Zhang and Jiang, BMC Bioinformatics 11:10, 2010) was made before our studies, the differences being that msoar first infers gene families based on similarities and then computes a matching based on a heuristic including structural rearrangements and tandem duplications, while FF-DCJ-Indel takes similarities and rearrangements simultaneously into account for inferring an optimal matching, in a rearrangement model including DCJ and mimicking all content modifications with insertions and deletions of DNA segments. The tool msoar was not maintained and is no longer operational, therefore we could never compare its performance to FF-DCJ-Indel.

[2] Our method was originally called DiffMGC [13], here renamed to OrthoFFGC.

Orthology inference via family-free genome rearrangements

For studying large-scale genome rearrangements a high-level view of a *chromosome* is adopted. In this view each chromosome is represented by a sequence of *genes*. Since each gene is an oriented DNA fragment, we need to distinguish its two possible representations: a gene g is represented by the symbol g itself, if it is read in direct orientation, or by the symbol $\overline{g}$, if it is read in reverse orientation. In our notation, all genes of a linear chromosome are concatenated in a string that can be read in any of the two directions and is flanked by square brackets. As an example, let $C = [\ \overline{6}\ 1\ 8\ 9\ \overline{4}\ ]$ be a linear chromosome. A *genome* is then a set of chromosomes and can be transformed with the following types of mutations:

1. **Structural rearrangements (DCJ operations):** A *cut* performed on a chromosome C of a genome $\mathbb{A}$ separates two adjacent genes of C. A *double-cut and join* or *DCJ* applied on genome $\mathbb{A}$ is the operation that performs cuts in two different positions of distinct chromosomes or of the same chromosome of $\mathbb{A}$, creating four open ends, and joins these open ends in a different way [1, 9]. For example, let $\mathbb{A} = \{ [\ \overline{6}\ 1\ 8\ 9\ \overline{4}\ ], [\ 3\ \overline{5}\ \overline{7}\ 2\ ] \}$, and consider a DCJ that cuts between genes 1 and 8 of its first chromosome and between genes 7 and 2 of its second chromosome, creating segments $\overline{6}\ 1\bullet$, $\bullet 8\ 9\ \overline{4}$, $3\ \overline{5}\ \overline{7}\bullet$ and $\bullet 2$ (where the symbols $\bullet$ represent the open ends). If we join the first with the fourth and the second with the third open end, we get $\mathbb{A}' = \{ [\ \overline{6}\ 1\ 2\ ], [\ 3\ \overline{5}\ \overline{7}\ 8\ 9\ \overline{4}\ ] \}$, that is, the described DCJ operation is a translocation transforming $\mathbb{A}$ into $\mathbb{A}'$. Indeed, a DCJ operation can correspond not only to a translocation but to several structural rearrangements, such as an inversion, a fusion or a fission.
2. **Content-modifying (indel operations):** The content of a chromosome can be modified with *insertions* and with *deletions* of blocks of contiguous genes, collectively called *indel* operations. Note that at most one chromosome can be entirely deleted or inserted at once. As an example, consider the deletion of segment $\overline{7}\ 8\ 9$ from chromosome $[\ 3\ \overline{5}\ \overline{7}\ 8\ 9\ \overline{4}\ ]$, resulting in chromosome $[\ 3\ \overline{5}\ \overline{4}\ ]$. A gene cannot be deleted and then reinserted, nor inserted and then deleted. This restriction prevents the *free lunch* artifact of sorting one genome into the other by simply deleting the chromosomes of the first and inserting the chromosomes of the second, ignoring their common parts.

Computing an optimal set of orthologs between two genomes

We can represent the pairwise similarities between the genes of genome $\mathbb{A}$ and the genes of genome $\mathbb{B}$ in the so called *gene similarity graph* [10], denoted by $\mathcal{S}(\mathbb{A}, \mathbb{B})$. This is a weighted bipartite graph that has a vertex for each gene in genome $\mathbb{A}$ and a vertex for each gene in genome $\mathbb{B}$. Furthermore, for each pair of genes $g_1 \in \mathbb{A}, g_2 \in \mathbb{B}$, denote by $\sigma(g_1, g_2)$ their *normalized similarity*, a value that ranges in the interval [0, 1]. Given a filter $\mathcal{F}$, if $\sigma(g_1, g_2)$ is not discarded by $\mathcal{F}$, there is an edge e connecting g_1 and g_2 in $\mathcal{S}(\mathbb{A}, \mathbb{B})$ whose score is $\sigma(e) = \sigma(g_1, g_2)$. In addition, to each vertex u of $\mathcal{S}(\mathbb{A}, \mathbb{B})$ we assign a weight $w(u)$ that can be obtained as follows: $w(u) = \max\{\sigma(uv) \mid uv \in \mathcal{S}(\mathbb{A}, \mathbb{B})\}$, that is, $w(u)$ is the maximum similarity among the edges incident to the vertex (or gene) u in $\mathcal{S}(\mathbb{A}, \mathbb{B})$.

A matching $\mathcal{O}$ from $\mathcal{S}(\mathbb{A}, \mathbb{B})$, here also called an *ortholog-set*, defines the tuple $(\mathbb{A}, \mathbb{B}, \mathcal{O})$, in which every two genes a, b, such that $a \in \mathbb{A}$, $b \in \mathbb{B}$ and $ab \in \mathcal{O}$, are considered to be *orthologs*. The *complement* of $\mathcal{O}$, denoted by $\widetilde{\mathcal{O}}$, is the set composed of genes whose corresponding vertices in $\mathcal{S}(\mathbb{A}, \mathbb{B})$ are $\mathcal{O}$-unsaturated.

The DCJ-indel distance $\mathrm{d}^{\mathrm{ID}}_{\mathrm{DCJ}}(\mathbb{A}, \mathbb{B}, \mathcal{O})$ is the minimum number of DCJ and indel operations required to transform $\mathbb{A}$ into $\mathbb{B}$ assuming the orthologs given by $\mathcal{O}$ and allowing only the genes belonging to the complement $\widetilde{\mathcal{O}}$ to be inserted or deleted. It can be computed using an approach relying on the cycles and paths of a graph that represents the structural relation between genomes $\mathbb{A}$ and $\mathbb{B}$ according to the ortholog-set $\mathcal{O}$ [3, 12] (this graph is equivalent to a consistent decomposition of the family-free relational graph, described in the next subsection and represented in Fig. 1 (bottom)). Together with the scores of edges and vertices of $\mathcal{S}(\mathbb{A}, \mathbb{B})$, the DCJ-indel distance $\mathrm{d}^{\mathrm{ID}}_{\mathrm{DCJ}}$ allows the computation of the weighted rearrangement distance $\mathrm{wd}^{\mathrm{ID}}_{\mathrm{DCJ}}$ [12]:

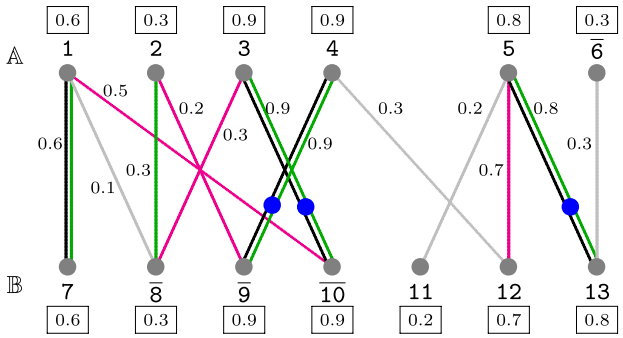
$$\mathrm{wd}^{\mathrm{ID}}_{\mathrm{DCJ}}(\mathbb{A}, \mathbb{B}, \mathcal{S}, \mathcal{O}) = \mathrm{d}^{\mathrm{ID}}_{\mathrm{DCJ}}(\mathbb{A}, \mathbb{B}, \mathcal{O}) + |\mathcal{O}| - \sigma(\mathcal{O}) + \mathrm{w}(\widetilde{\mathcal{O}}),$$

where

$$\sigma(\mathcal{O}) = \sum_{e \in \mathcal{O}} \sigma(e) \text{ and } w(\widetilde{\mathcal{O}}) = \sum_{v \in \widetilde{\mathcal{O}}} w(v).$$

Then, given that $\mathfrak{M}$ is the set of all possible ortholog-sets in $\mathcal{S}(\mathbb{A}, \mathbb{B})$, the rearrangement distance between $\mathbb{A}$ and $\mathbb{B}$ is the result of the following optimization:

$$\textsc{GenDiff}(\mathbb{A}, \mathbb{B}, \mathcal{S}) = \min_{\mathcal{O} \in \mathfrak{M}} \{\mathrm{wd}^{\mathrm{ID}}_{\mathrm{DCJ}}(\mathbb{A}, \mathbb{B}, \mathcal{S}, \mathcal{O})\}.$$



Ranking of the represented ortholog-sets based on their corresponding distances

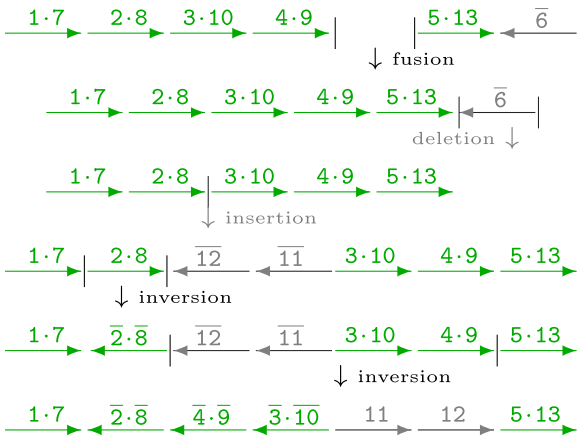
$\mathcal{O}$	$\lvert\mathcal{O}\rvert$	$\sigma(\mathcal{O})$	$w(\widetilde{\mathcal{O}})$	$\mathrm{d}_{\mathrm{DCJ}}^{\mathrm{ID}}$	$\mathrm{wd}_{\mathrm{DCJ}}^{\mathrm{ID}}$
black	4	3.2	1.8	5 *	7.6
green	5	3.5	1.2	5 *	7.7
blue	3	2.6	3.0	5	8.4
magenta	4	1.7	2.8	6	11.1

* Rearrangement scenarios are given below, with genes belonging to $\widetilde{\mathcal{O}}$ as well as indel operations represented in gray

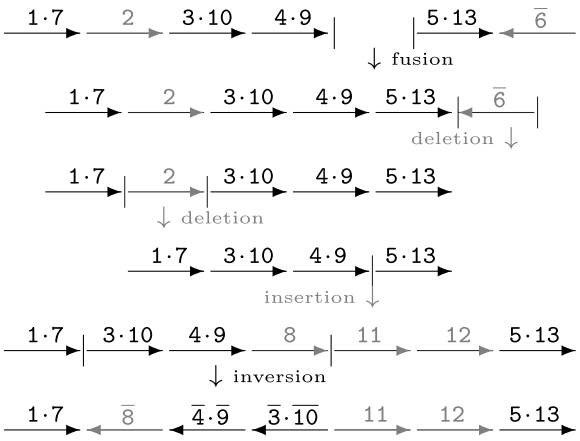
black ⊂ green (black lines next to green lines)

blue ⊂ black (blue dots on green/black lines)

green ortholog-set ($\mathrm{d}_{\mathrm{DCJ}}^{\mathrm{ID}} = 5$):



black ortholog-set ($\mathrm{d}_{\mathrm{DCJ}}^{\mathrm{ID}} = 5$):



$FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ and decomposition corresponding to the black ortholog-set of $\mathcal{S}(\mathbb{A},\mathbb{B})$ shown on the top-left corner

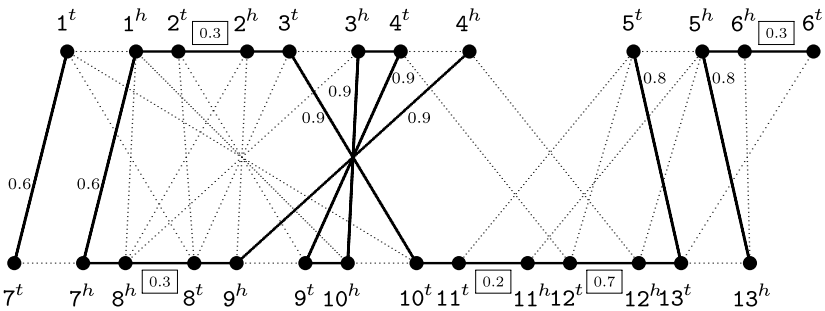


Fig. 1 On the top part is displayed the gene similarity graph $\mathcal{S}(\mathbb{A},\mathbb{B})$ of genomes $\mathbb{A} = \{[\ 1\ \ 2\ \ 3\ \ 4\ ][\ 5\ \ \overline{6}\ ]\}$ and $\mathbb{B} = \{[\ 7\ \ \overline{8}\ \ \overline{9}\ \ \overline{10}\ \ 11\ \ 12\ \ 13\ ]\}$ and next to it a table with the ranking of four distinct ortholog-sets. On the middle the rearrangement scenarios induced by two of these ortholog-sets are shown. On the bottom part the family-free relational graph $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ is illustrated, highlighting the edges of the decomposition corresponding to the (black) ortholog-set $\mathcal{O} = \{\{1,7\},\{3,10\},\{4,9\},\{5,13\}\}$. (This decomposition has two $\mathbb{AB}$-paths, one $\mathbb{AA}$-path and one cycle.) All extremity and indel edges in $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ are respectively scored and weighted according to $\mathcal{S}(\mathbb{A},\mathbb{B})$ but the scores and weights of edges not derived from $\mathcal{O}$ or $\widetilde{\mathcal{O}}$ are omitted

Figure 1 shows examples of ortholog-sets and their distances. Denote by $\textsc{OrthoFF}(\mathbb{A},\mathbb{B},\mathcal{S})$ an *optimal* ortholog-set in $\mathcal{S}(\mathbb{A},\mathbb{B})$, which is an ortholog-set whose rearrangement distance equals $\textsc{GenDiFF}(\mathbb{A},\mathbb{B},\mathcal{S})$. Computing the rearrangement distance $\textsc{GenDiFF}(\mathbb{A},\mathbb{B},\mathcal{S})$ and finding an optimal ortholog-set $\textsc{OrthoFF}(\mathbb{A},\mathbb{B},\mathcal{S})$ are NP-hard problems [12].

Family-free relational graph

For solving both NP-hard problems $\textsc{GenDiFF}(\mathbb{A},\mathbb{B},\mathcal{S})$ and $\textsc{OrthoFF}(\mathbb{A},\mathbb{B},\mathcal{S})$ we adopt the approach of decomposing the following graph.

The *family-free relational graph* $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$, shown in Fig. 1 (bottom), represents all possible weighted distances corresponding to all candidate ortholog-sets in $\mathcal{S}(\mathbb{A},\mathbb{B})$ [12]. Given a gene m, denote the extremities of m by m^h (*head*) and m^t (*tail*). The graph $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ has a set $V(\mathbb{A})$ with a vertex for each of the two extremities of each gene of genome $\mathbb{A}$ and a set $V(\mathbb{B})$ with a vertex for each of the two extremities of each gene of genome $\mathbb{B}$.

The set of edges is partitioned into several subsets:

- Sets $E^{\mathbb{A}}_{\text{adj}}$ and $E^{\mathbb{B}}_{\text{adj}}$ contain adjacency edges connecting adjacent extremities of genes in $\mathbb{A}$ and in $\mathbb{B}$.
- The set E_γ contains, for each edge $ab \in \mathcal{S}(\mathbb{A},\mathbb{B})$, an extremity edge connecting a^t to b^t, and an extremity edge connecting a^h to b^h. To both edges a^tb^t and a^hb^h, that are called *siblings*, we assign the same score $\sigma(ab)$ of the edge ab in $\mathcal{S}(\mathbb{A},\mathbb{B})$: $\sigma(a^tb^t) = \sigma(a^hb^h) = \sigma(ab)$.
- Sets $E^{\mathbb{A}}_{\text{id}}$ and $E^{\mathbb{B}}_{\text{id}}$ contain indel edges connecting the two extremities of each gene in $\mathbb{A}$ and in $\mathbb{B}$. Each indel edge m^hm^t receives the weight $w(m)$ of vertex m in $\mathcal{S}(\mathbb{A},\mathbb{B})$: $w(m^hm^t) = w(m)$. (Recall that $w(m)$ is the maximum score among the edges incident to m in $\mathcal{S}(\mathbb{A},\mathbb{B})$.)

Consistent decompositions of the family-free relational graph

A *decomposition* of $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ is a collection of edges building vertex-disjoint components, that can be cycles and/or paths, covering all vertices of $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$. A decomposition must be *consistent*, implying that it is *induced* by a set of edges $\mathcal{L}$ exclusively composed of pairs of siblings and without any pair of incident edges. The set $\mathcal{L}$ is called a *sibling-set* and corresponds to one precise ortholog-set of $\mathcal{S}(\mathbb{A},\mathbb{B})$, denoted by $\mathcal{O}_\mathcal{L}$. Note that $|\mathcal{L}| = 2|\mathcal{O}_\mathcal{L}|$ and $\sigma(\mathcal{L}) = 2\sigma(\mathcal{O}_\mathcal{L})$. The *complement* of $\mathcal{L}$, denoted by $\widetilde{\mathcal{L}}$, is composed of the indel-edges corresponding to the genes of $\widetilde{\mathcal{O}}_\mathcal{L}$ (the complement of $\mathcal{O}_\mathcal{L}$), therefore $|\widetilde{\mathcal{L}}| = |\widetilde{\mathcal{O}}_\mathcal{L}|$ and $w(\widetilde{\mathcal{L}}) = w(\widetilde{\mathcal{O}}_\mathcal{L})$. The consistent decomposition induced by $\mathcal{L}$ is denoted by $D[\mathcal{L}]$ and corresponds to:

$$D[\mathcal{L}] = \mathcal{L} \cup \widetilde{\mathcal{L}} \cup E^{\mathbb{A}}_{\text{adj}} \cup E^{\mathbb{B}}_{\text{adj}}.$$

The consistent decomposition $D[\mathcal{L}]$ covers all vertices of $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ and is composed of cycles and paths. The paths connect the ends of linear chromosomes in both genomes and can be of three types: either $\mathbb{A}\mathbb{A}$-path, or $\mathbb{B}\mathbb{B}$-path or $\mathbb{A}\mathbb{B}$-path.

The structure of $D[\mathcal{L}]$ has all necessary information for computing the value $\text{wd}^{\text{ID}}_{\text{DCJ}}(\mathbb{A},\mathbb{B},\mathcal{S},\mathcal{O}_\mathcal{L})$, therefore we can say that $\text{wd}^{\text{ID}}_{\text{DCJ}}(\mathbb{A},\mathbb{B},\mathcal{S},\mathcal{O}_\mathcal{L}) = \text{wd}^{\text{ID}}_{\text{DCJ}}(\text{D}[\mathcal{L}])$ [12]. Given that $\mathfrak{S}$ is the set of all possible sibling-sets in $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$, we can modify our optimization problem to

$$\textsc{GenDiFF}(\mathbb{A},\mathbb{B},\mathcal{S}) = \min_{\mathcal{L}\in\mathfrak{S}}\{\text{wd}^{\text{ID}}_{\text{DCJ}}(\text{D}[\mathcal{L}])\}.$$

Assuming that a sibling-set $\mathcal{L}_\star$ gives the optimal solution for the problem $\textsc{GenDiFF}(\mathbb{A},\mathbb{B},\mathcal{S})$, then $\textsc{OrthoFF}(\mathbb{A},\mathbb{B},\mathcal{S}) = \mathcal{O}_{\mathcal{L}_\star}$.

Capping

The end of a linear chromosome is called *telomere*. The telomeres are also the ends of the paths of any consistent decomposition. Therefore, if $\kappa(\mathbb{A})$ is the number of linear chromosomes in $\mathbb{A}$ and $\kappa(\mathbb{B})$ is the number of linear chromosomes in $\mathbb{B}$ the number of paths in any decomposition is $\kappa(\mathbb{A}) + \kappa(\mathbb{B})$. Our ILP is able to capture all necessary properties from the cycles of a decomposition, but cannot handle paths. A way to overcome this problem is by linking all paths of any decomposition with a known technique called *capping* [2].

Capping a consistent decomposition

The idea of the capping is to split the telomeres into disjoint pairs and then to connect the two elements of each pair, so that all paths are linked into cycles. The only restriction is that a pair cannot contain telomeres from the same genome, therefore, if the numbers of telomeres in the two genomes are different, some *dummy* telomeres need to be created, as we describe in the following.

Suppose that $D[\mathcal{L}]$ is any consistent decomposition of $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$. For each telomere (vertex) v, add to $D[\mathcal{L}]$ a *cap vertex* θ_v and connect v to θ_v by an adjacency edge. Now let $\theta(\mathbb{A})$ (respectively $\theta(\mathbb{B})$) be the set of all cap vertices in $\mathbb{A}$ (respectively in $\mathbb{B}$). Note that, since each linear chromosome has two ends, the cardinalities of these sets must be even. Moreover, if $|\theta(\mathbb{A})| \neq |\theta(\mathbb{B})|$, the cardinalities of these sets must be equalized. Let $p_* = \max\{\kappa(\mathbb{A}),\kappa(\mathbb{B})\}$ and $a_* = |\kappa(\mathbb{A}) - \kappa(\mathbb{B})|$. For

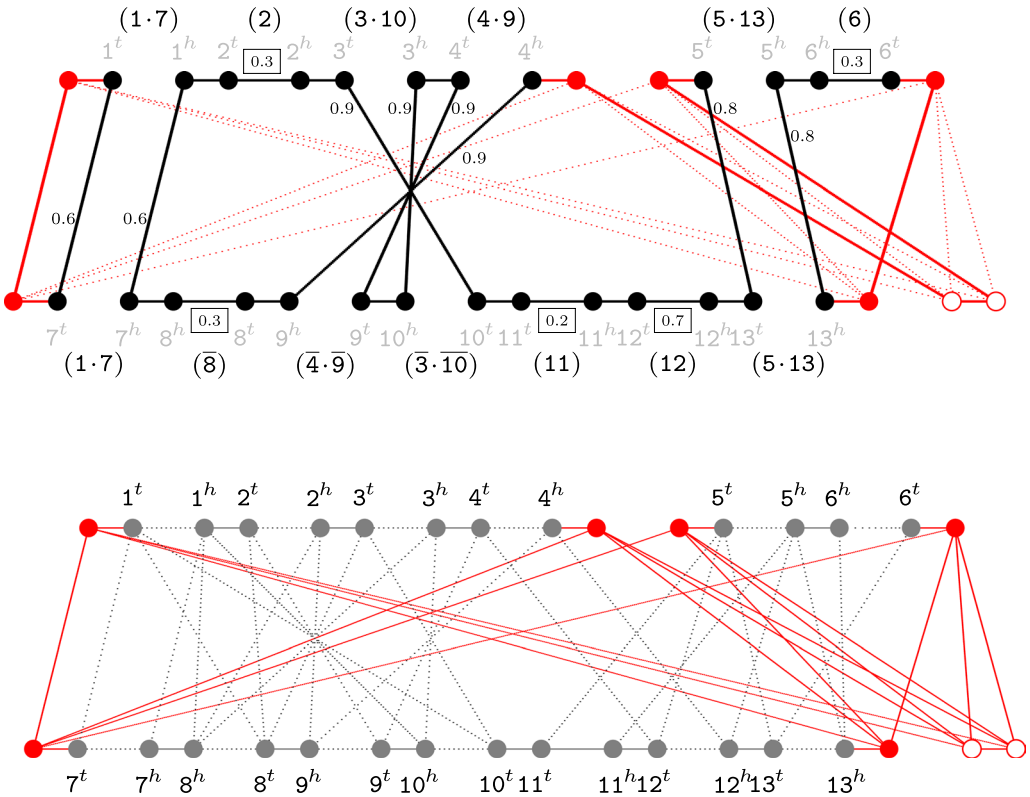


Fig. 2 On the top part we show the capping of the decomposition corresponding to the (`black`) ortholog-set $\mathcal{O} = \{\{1,7\},\{3,10\},\{4,9\},\{5,13\}\}$ from the gene similarity graph $\mathcal{S}(\mathbb{A},\mathbb{B})$ of Fig. 1 (bottom). Each red vertex is a cap vertex. Each filled (red) vertex is connected to a telomere (chromosome/path ends). The unfilled vertices represent the extra (equalizing) vertices connected by a dummy adjacency. The capping is a perfect matching of the complete bipartite graph of the cap vertices. The optimal capping for this decomposition is highlighted. It closes each of its paths into a separate cycle. (In general, an optimal capping of a decomposition may link up to 4 paths into a single cycle [8]). On the bottom part is displayed the complete family-free graph $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ optimally capped. Cap edges are unweighted. Scores of extremity edges and weights of indel edges are omitted

equalizing the cardinalities with the minimum number of extra vertices, we need to add $2a_*$ extra cap vertices to the set with smaller cardinality. These extra cap vertices must be split into pairs (arbitrarily chosen) so that the vertices of each pair are connected by a *dummy* adjacency edge in $D[\mathcal{L}]$. Denote by $\hat{\theta}(\mathbb{A})$ and $\hat{\theta}(\mathbb{B})$ the sets with equalized cardinalities and let P be a *capping-set*, which is a perfect matching between them: for $\gamma \in \hat{\theta}(\mathbb{A})$ and $\gamma' \in \hat{\theta}(\mathbb{B})$, if $\gamma\gamma' \in P$, then γ and γ' are connected by a *cap edge*. Let $\widehat{D}[\mathcal{L},P]$ be a *capped decomposition* of $D[\mathcal{L}]$ with capping-set P. It is easy to see that $\widehat{D}[\mathcal{L},P]$ is composed of cycles only.

Optimal capping

So far we explained how to guarantee that all paths in any decomposition are linked into cycles. Note, however, that there are $(2p_*)!$ ways of completely matching the vertices of sets $\hat{\theta}(\mathbb{A})$ and $\hat{\theta}(\mathbb{B})$. For a given decomposition $D[\mathcal{L}]$, any of these possibilities, say capping-set P, would produce a capped decomposition $\widehat{D}[\mathcal{L},P]$, and capping the same $D[\mathcal{L}]$ with distinct capping-sets may produce distinct weighted costs. Let $P_{\mathcal{L}}$ be an optimal capping-set for $D[\mathcal{L}]$, that is, $\widehat{D}[\mathcal{L},P_{\mathcal{L}}]$ has the minimum weighted cost among all cappings of $D[\mathcal{L}]$.[3] In a previous work we have shown that $\mathrm{wd}^{\mathrm{ID}}_{\mathrm{DCJ}}(\widehat{\mathrm{D}}[\mathcal{L},\mathrm{P}_{\mathcal{L}}]) = \mathrm{wd}^{\mathrm{ID}}_{\mathrm{DCJ}}(\mathrm{D}[\mathcal{L}])$ [12]. Therefore, for a consistent decomposition $D[\mathcal{L}]$, any of its optimal capping-sets preserves the weighted cost of $D[\mathcal{L}]$, reporting both $\mathrm{d}^{\mathrm{ID}}_{\mathrm{DCJ}}(\mathbb{A},\mathbb{B},\mathcal{O}_{\mathcal{L}})$ and $\mathrm{wd}^{\mathrm{ID}}_{\mathrm{DCJ}}(\mathbb{A},\mathbb{B},\mathcal{S},\mathcal{O}_{\mathcal{L}})$. Figure 2 (top) highlights an optimal capping of a consistent decomposition.

[3] There can be several co-optimal capping-sets for the same decomposition $D[\mathcal{L}]$ and each optimal capping-set links up to 4 cycles of $D[\mathcal{L}]$ into a single cycle [8].

Optimally capped family-free relational graph

All consistent decompositions share the same telomeres, therefore a set of capping-sets for one decomposition is also a set of capping-sets of any other decomposition. If we then simply add all possible capping-sets to the family-free relational graph, which implies adding a complete bipartite graph with partite sets $\hat{\theta}(\mathbb{A})$ and $\hat{\theta}(\mathbb{B})$, we guarantee that an optimal solution can be found. Let the so-called optimal capping (represented in Fig. 2 (bottom)) of $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ with the minimum number of extra elements be denoted by $\theta_\star(FFR(\mathbb{A},\mathbb{B},\mathcal{S}))$ and be defined as follows:

- ($1_\star$) Add the set of *cap vertices* $\hat{\theta}(\mathbb{A}) = \theta^1_{\mathbb{A}}, \theta^2_{\mathbb{A}}, \ldots, \theta^{2p_*}_{\mathbb{A}}$ and connect each telomere of genome $\mathbb{A}$ to one of these cap vertices by an adjacency edge added to $E^{\mathbb{A}}_{\text{adj}}$.
- ($2_\star$) Similarly, add the set of cap vertices $\hat{\theta}(\mathbb{B}) = \theta^1_{\mathbb{B}}, \theta^2_{\mathbb{B}}, \ldots, \theta^{2p_*}_{\mathbb{B}}$ and connect each telomere of genome $\mathbb{B}$ to one of these cap vertices by an adjacency edge added to $E^{\mathbb{B}}_{\text{adj}}$.
- ($3_\star$) Add (arbitrarily chosen) $p_* - \kappa(\mathbb{A})$ dummy adjacency edges to $E^{\mathbb{A}}_{\text{adj}}$ and $p_* - \kappa(\mathbb{B})$ dummy adjacency edges to $E^{\mathbb{B}}_{\text{adj}}$. (Note that only one of the two genomes may have dummy adjacencies.)
- ($4_\star$) Connect all cap vertices in $\hat{\theta}(\mathbb{A})$ to all cap vertices in $\hat{\theta}(\mathbb{B})$ with *cap edges*. The set of all cap edges is denoted by E_θ.

Since all $2p_*$ cap vertices in $\mathbb{A}$ are connected to all $2p_*$ cap vertices in $\mathbb{B}$ and any perfect matching of these edges is a valid capping, the search space of our optimization problem is multiplied by $(2p_*)!$. Denote by $\mathfrak{P}$ the set of all possible capping-sets (perfect matchings) between the vertices from $\hat{\theta}(\mathbb{A})$ and $\hat{\theta}(\mathbb{B})$. The optimization problem over $\theta_\star(FFR(\mathbb{A},\mathbb{B},\mathcal{S}))$ can be rewritten as

$$\textsc{GenDiFF}(\mathbb{A},\mathbb{B},\mathcal{S}) = \min_{\mathcal{L}\in\mathfrak{S},P\in\mathfrak{P}} \{\text{wd}^{\text{ID}}_{\text{DCJ}}(\widehat{\text{D}}[\mathcal{L},\text{P}]\}.$$

Assuming that a sibling-set $\mathcal{L}_\star$ (together with one of its optimal capping-sets) gives the optimal solution for $\textsc{GenDiFF}(\mathbb{A},\mathbb{B},\mathcal{S})$, an optimal ortholog-set is $\textsc{OrthoFF}(\mathbb{A},\mathbb{B},\mathcal{S}) = \mathcal{O}_{\mathcal{L}_\star}$. Both problems GenDiFF and OrthoFF can be solved with the ILP formulation FF-DCJ-Indel [12], which can be found in Additional file 1: Section (A).

Integration of pairwise optimal ortholog-sets into gene families

In our previous work [13], the ILP FF-DCJ-Indel solving OrthoFF (with optimal capping) was integrated in a tool called OrthoFFGC for inferring gene families across several species. The pipeline can be summarized as follows: given a set of n genomes, gene similarities and ortholog-sets are computed for all pairwise comparisons and simply integrated into an n-partite graph. The connected components of this graph are the inferred gene families.

In this work we modify this pipeline by replacing the optimal capping by a heuristic lighter one, as we explain in the next section.

Heuristic capping

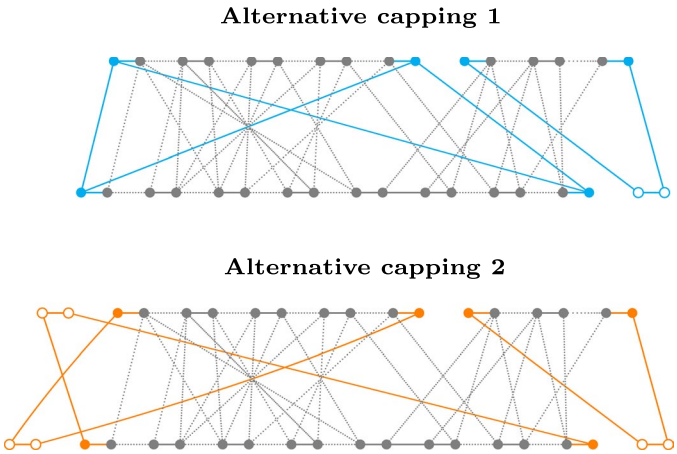
Conceptually our approach can handle genomes with several linear chromosomes and even partially assembled genomes distributed into many contigs/scaffolds: each of these is a linear segment and could simply be treated as the same object that we so far called chromosome. However, as already explained, the optimal capping multiplies the search space of $FFR(\mathbb{A},\mathbb{B})$ by $(2p_*)!$ where p_* is the maximum between the number of linear segments in genomes $\mathbb{A}$ and $\mathbb{B}$. This effect makes it unfeasible to analyze genomes with a large number of segments with our ILP over an optimally capped family-free relational graph.

One way of overcoming this issue is by adopting a lighter capping, for example by removing some edges from the complete bipartite graph of $\hat{\theta}(\mathbb{A})$ and $\hat{\theta}(\mathbb{B})$, and/or by partitioning these sets into subsets that are capped independently. In any case it is important to guarantee that a capping is *valid*, that is, it allows to find a capping-set (a perfect matching of the cap vertices). A valid lighter capping may not include the optimal capping-sets, and therefore may not preserve the computed weighted costs. Note, however, that even if the weighted costs are not preserved, the ranking of the ortholog-sets/sibling-sets may not be affected. In Fig. 3 we show examples of arbitrary lighter valid cappings and their effects on the ortholog-set ranking.

Perfect shared-content graph with thresholds τ and ϵ

Our goal is therefore to develop a lighter heuristic capping that may potentially preserve the original (optimal) ranking of the best ortholog-sets/sibling-sets. We achieve this by connecting cap vertices only between the telomeres of the linear segments (contigs or chromosomes) that (potentially) share most of their genomic contents. This is because those telomeres have a higher chance of being in the same paths of the best consistent decompositions of the family-free relational graph.

Given two linear segments $A \in \mathbb{A}$ and $B \in \mathbb{B}$, let their *shared genomic content* $\Omega(A,B)$ be the sum of the scores



Orthologs-sets and their corresponding costs with the alternative lighter cappings above

				Optimal		Alternative 1		Alternative 2	
$\mathcal{O}$	$\lvert\mathcal{O}\rvert$	$\sigma(\mathcal{O})$	$w(\widetilde{\mathcal{O}})$	$\mathrm{d}_{\mathrm{DCJ}}^{\mathrm{ID}}$	$\mathrm{wd}_{\mathrm{DCJ}}^{\mathrm{ID}}$	$\mathrm{d}_{\mathrm{DCJ}}^{\mathrm{ID}}$	$\mathrm{wd}_{\mathrm{DCJ}}^{\mathrm{ID}}$	$\mathrm{d}_{\mathrm{DCJ}}^{\mathrm{ID}}$	$\mathrm{wd}_{\mathrm{DCJ}}^{\mathrm{ID}}$
black	4	3.2	1.8	5	7.6	6	8.6	8	10.6
green	5	3.5	1.2	5	7.7	6	8.7	8	10.7
blue	3	2.6	3.0	5	8.4	6	9.4	7	10.4
magenta	4	1.7	2.8	6	11.1	6	11.1	7	12.1

Fig. 3 Examples of two arbitrary lighter valid cappings of the family-free relational diagram from Fig. 1 (bottom) and their effects on the ranking of ortholog-sets/sibling-sets represented in Fig. 1 (top). Both cappings affect the computed distances, but, while the capping shown in the left (cyan) preserves the optimal ranking, the one shown in the right (orange) does not

of edges in $\mathcal{S}$ connecting genes from A to genes from B. Formally, for $A \subseteq \mathbb{A}$, $B \subseteq \mathbb{B}$, we have

$$\Omega(A,B) = \sum_{\substack{g_1g_2 \in \mathcal{S} \\ g_1 \in A \\ g_2 \in B}} \sigma(g_1g_2).$$

Now let $\mathcal{C}(\mathbb{A},\mathbb{B}) = (U_{\mathbb{A}}, U_{\mathbb{B}}, F)$ be the bipartite *shared-content graph* where the vertex sets are

$$U_{\mathbb{A}} = \{A : A \text{ is a linear segment in } \mathbb{A}\} \text{ and}$$
$$U_{\mathbb{B}} = \{B : B \text{ is a linear segment in } \mathbb{B}\} .$$

Initially the set of edges is $F = \{AB \mid A \in U_{\mathbb{A}}, B \in U_{\mathbb{B}} \text{ and } \Omega(A,B) > 0\}$. Each edge AB has a score $\Omega(A,B)$. We then reduce the size of $\mathcal{C}(\mathbb{A},\mathbb{B}) = (U_{\mathbb{A}}, U_{\mathbb{B}}, F)$, by removing edges based on two parameters:

(i) Given a positive integer τ, we remove edges from F by applying a filtering procedure that simply iterates over $U_{\mathbb{A}} \cup U_{\mathbb{B}}$, keeping in F only the τ edges of highest scores for each vertex.

(ii) Then, given a rational threshold $\epsilon \in (0,1]$, the remaining edges are again filtered out to remove weak relations between linear segments: let the score of a vertex $v \in \mathcal{C}(\mathbb{A},\mathbb{B})$ be $\Omega(v) = \max_{uv \in F}\{\Omega(u,v)\}$; the edges inciding on v that have scores below $\epsilon\ \Omega(v)$ are removed.

Capping attempt induced by the shared-content graph

The shared-content graph will now *induce* our capping procedure. The idea is to allow cap connections only between the ends of linear segments that are connected in $\mathcal{C}$. Therefore, the linear segments that are in the same connected component of $\mathcal{C}$ will be capped together, independently from the linear segments that are in other connected components. In other words, the connected components of $\mathcal{C}$ will impose a partitioning of the capping procedure.

Let us then assume that $\mathcal{C}$ has a single connected component. Note that a capping induced by $\mathcal{C}$ can only be valid if its partite sets $U_{\mathbb{A}}$ and $U_{\mathbb{B}}$ are of the same size. This necessary condition also applies and is sufficient for the optimal capping, but here it is not sufficient: even when $U_{\mathbb{A}}$ and $U_{\mathbb{B}}$ are of the same size, since not all connections between the ends of the linear segments in $U_{\mathbb{A}}$ and in $U_{\mathbb{B}}$ are present, the induced capping could be invalid (Fig. 4a). Here the necessary and sufficient condition for a valid capping is

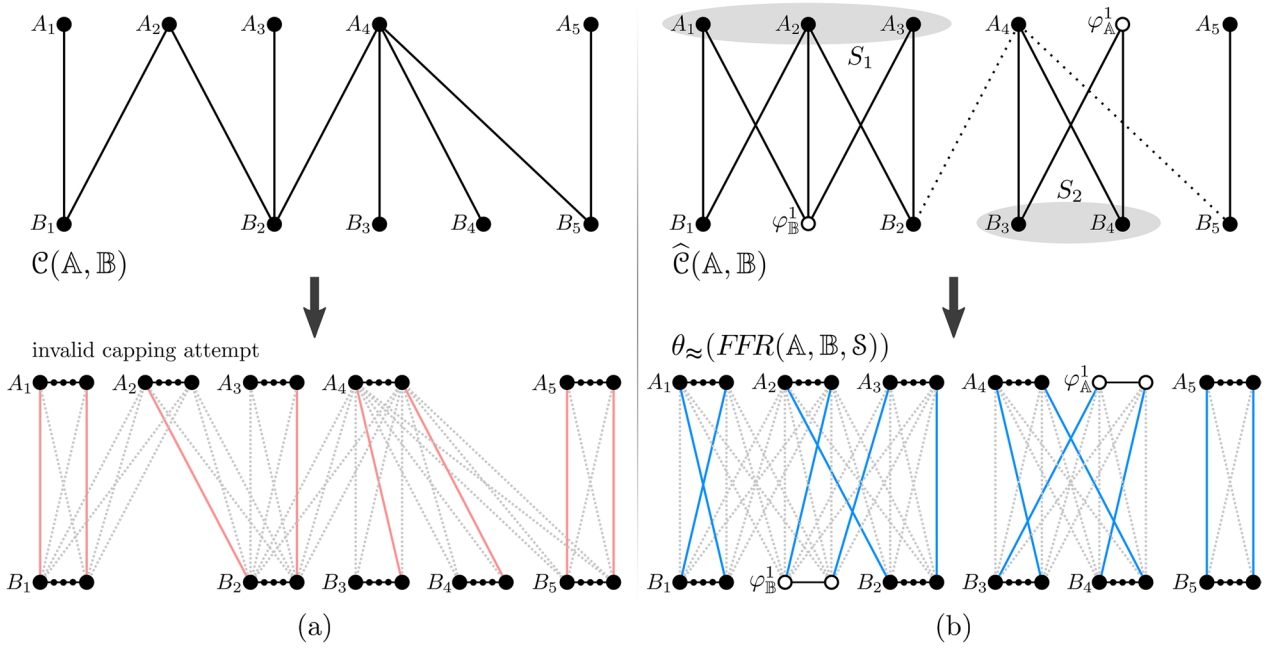


Fig. 4 **a** Example of a shared-content graph $\mathcal{C}(\mathbb{A},\mathbb{B})$ with omitted edge scores. The genomes $\mathbb{A}$ and $\mathbb{B}$ have linear segments A_{1_5} and B_{1_5}, respectively. The capping of $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ induced by $\mathcal{C}$ is invalid. **b** Transformation of $\mathcal{C}$ into a perfect shared-content graph $\widehat{\mathcal{C}}(\mathbb{A},\mathbb{B})$: vertex sets S_1 and S_2 represent Hall violators (among other possibilities) that demand the creation of dummy segments $\varphi^1_{\mathbb{B}}$ and $\varphi^1_{\mathbb{A}}$, respectively. Dotted edges represent those that are non-matchable and must be removed from $\widehat{\mathcal{C}}$ after the completion is finished. Notice that the component with vertices $A_1, A_2, A_3, B_1, \varphi^1_{\mathbb{B}}$ and B_2 is not a complete bipartite subgraph. (In both (**a**, **b**), we give an abstract illustration of the capped *FFR* where only cap vertices, cap edges and dummy adjacencies are represented explicitly, while vertices of gene extremities between cap vertices are represented by a line with small dots. In addition, colored solid edges represent a maximum cardinality matching between cap vertices, while the cap edges not in the matching are dashed grey)

the existence of a perfect matching in $\mathcal{C}(\mathbb{A},\mathbb{B})$, as stated in Lemma 1, whose proof relies on a theorem closely related to perfect matchings and demonstrated by Hall [20] in 1935. Denote by $\mathcal{N}(S)$ the neighborhood of a vertex set S, that is, the set of all vertices adjacent to some vertex of S.

Theorem 1 **(Hall's marriage theorem)** *Let $G = (U, V, E)$ be a bipartite graph. There exists a matching in G that covers the vertex set V if and only if for each subset $S \subseteq V$, $|S| \leq |\mathcal{N}(S)|$.*

Note that a perfect matching exists in $\mathcal{C}$ if and only if the condition of Theorem 1 holds for both $V_{\mathbb{A}}$ and $V_{\mathbb{B}}$. We can now establish the relation between perfect matchings in $\mathcal{C}$ and the validity of the capping it induces in the family-free relational graph.

Lemma 1 *A perfect matching exists in $\mathcal{C}(\mathbb{A},\mathbb{B})$ if and only if the capping of $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ induced by $\mathcal{C}$ is valid.*

Proof *If a perfect matching M exists in $\mathcal{C}$, for each edge AB in M, in the induced capping of $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ the pair of cap vertices connected to the ends of A can be connected in any of the two distinct ways to the pair of cap vertices connected to the ends of B, resulting in a capping-set.*
The converse is shown by contraposition. Suppose that a maximum cardinality matching M in $\mathcal{C}$ is not a perfect matching. Therefore, by Hall's marriage theorem, there exists some S in $\mathcal{C}$ such that $|\mathcal{N}(S)| < |S|$. Let S' be the set of cap vertices in the capping induced by $\mathcal{C}$ for all linear segments in S. Since the connection of these cap vertices follows $\mathcal{C}$ and each linear segment has two cap vertices, it is clear that $|S'| = 2|S|$ and $|\mathcal{N}(S')| = 2|\mathcal{N}(S)|$, hence, $|\mathcal{N}(S')| < |S'|$. By the pigeonhole principle, at least 2 cap vertices (because $|\mathcal{N}(S')|$ and $|S'|$ are even numbers) will not be incident to any cap edge, therefore no capping-set exists. □

Building the perfect shared-content graph

We will now describe a procedure for transforming the shared-content graph $\mathcal{C}(\mathbb{A},\mathbb{B})$ into a *perfect shared-content graph* $\widehat{\mathcal{C}}(\mathbb{A},\mathbb{B})$ that has at least one perfect matching, as shown in Algorithm 1. In the completion loop, dummy segments are iteratively created until a perfect matching is possible. If a maximum cardinality matching M is found but is not a perfect matching, a Hall violator set S can be found as follows. Let v be a vertex unsaturated by M, then $S = \{v\} \cup \{u \mid u$ is reachable from v by an M-alternating path$\}$. Finally, $|S| - |\mathcal{N}(S)|$ dummy segments are created and connected to each linear segment in S.

Algorithm 1 Transforms a shared-content graph into a perfect one

Input: A shared-content graph $\mathcal{C}(\mathbb{A},\mathbb{B}) = (U_\mathbb{A}, U_\mathbb{B}, F)$
Output: A perfect shared-content graph $\widehat{\mathcal{C}}(\mathbb{A},\mathbb{B}) = (\widehat{U}_\mathbb{A}, \widehat{U}_\mathbb{B}, \widehat{F})$

$\widehat{U}_\mathbb{A} \leftarrow U_\mathbb{A}$, $\widehat{U}_\mathbb{B} \leftarrow U_\mathbb{B}$, $\widehat{F} \leftarrow F$
while a maximum cardinality matching M in $\widehat{F}$ is not a perfect matching **do**
  $S \leftarrow$ a Hall violator set derived from M
  $\Phi \leftarrow$ dummy segments $\{\varphi^1, \ldots, \varphi^{|S|-|\mathcal{N}(S)|}\}$
  if $S \subseteq \widehat{U}_\mathbb{A}$ **then** $\widehat{U}_\mathbb{B} \leftarrow \widehat{U}_\mathbb{B} \cup \Phi$ **else** $\widehat{U}_\mathbb{A} \leftarrow \widehat{U}_\mathbb{A} \cup \Phi$
  $\widehat{F} \leftarrow \widehat{F} \cup (S \times \Phi)$
remove from $\widehat{F}$ all non-matchable edges
return $\widehat{\mathcal{C}} = (\widehat{U}_\mathbb{A}, \widehat{U}_\mathbb{B}, \widehat{F})$

An edge in $\widehat{\mathcal{C}}(\mathbb{A},\mathbb{B})$ is *matchable* if it is part of at least one perfect matching and *non-matchable* otherwise. Once the completion loop is finished, $\widehat{\mathcal{C}}$ admits at least one perfect matching and its matchable edges can be identified efficiently [21]. The last step of Algorithm 1 is then removing from $\widehat{\mathcal{C}}$ all non-matchable edges. An example of the construction of a perfect shared-content graph is illustrated above, in Fig. 4b.

Heuristically capped family-free relational graph

The bipartite perfect shared-content graph $\widehat{\mathcal{C}}(\mathbb{A},\mathbb{B})$ has edge set $\widehat{F}$ and partite sets $\widehat{U}_\mathbb{A} = U_\mathbb{A} \cup U^\varphi_\mathbb{A}$ and $\widehat{U}_\mathbb{B} = U_\mathbb{B} \cup U^\varphi_\mathbb{B}$, where $U_\mathbb{A}$ and $U_\mathbb{B}$ are the sets of linear segments and $U^\varphi_\mathbb{A}$ and $U^\varphi_\mathbb{B}$ the sets of dummy segments. Recall that the sets $\widehat{U}_\mathbb{A}$ and $\widehat{U}_\mathbb{B}$ have the same cardinality, which here we denote by $p_\approx$. The heuristic capping $\theta_\approx$ of the family-free relational graph $FFR(\mathbb{A},\mathbb{B},\mathcal{S})$ induced by $\widehat{\mathcal{C}}(\mathbb{A},\mathbb{B})$ is shown in Fig. 4b and described as follows:

($1_\approx$) Add the set of cap vertices $\hat{\theta}(\mathbb{A}) = \theta^1_\mathbb{A}, \theta^2_\mathbb{A}, \ldots, \theta^{2p_\approx}_\mathbb{A}$. For $i = 1 \ldots |U_\mathbb{A}|$, associate each linear segment $A_i \in U_\mathbb{A}$ to cap vertices $\theta^{2i-1}_\mathbb{A}$ and $\theta^{2i}_\mathbb{A}$ and connect with adjacency edges one telomere of A_i to $\theta^{2i-1}_\mathbb{A}$ and the other to $\theta^{2i}_\mathbb{A}$. Note that $2|U^\varphi_\mathbb{A}|$ cap vertices remain disconnected.

($2_\approx$) Similarly, add cap vertices $\hat{\theta}(\mathbb{B}) = \theta^1_\mathbb{B}, \theta^2_\mathbb{B}, \ldots, \theta^{2p_\approx}_\mathbb{B}$. For $j = 1 \ldots |V_\mathbb{B}|$, associate each linear segment $B_j \in U_\mathbb{B}$ to cap vertices $\theta^{2j-1}_\mathbb{B}$ and $\theta^{2j}_\mathbb{B}$ and connect with adjacency edges one telomere of B_j to $\theta^{2j-1}_\mathbb{B}$ and the other to $\theta^{2j}_\mathbb{B}$. Again, $2|U^\varphi_\mathbb{B}|$ cap vertices remain disconnected.

($3_\approx$) For $i_\circ = 1 \ldots |U^\varphi_\mathbb{A}|$ and $i = |U_\mathbb{A}| + i_\circ$, connect the pair of cap vertices $\theta^{2i-1}_\mathbb{A}$ and $\theta^{2i}_\mathbb{A}$ by a dummy adjacency edge, associating this pair to the dummy segment $\varphi^{i_\circ}_\mathbb{A} \in U^\varphi_\mathbb{A}$. Similarly, for $j_\circ = 1 \ldots |U^\varphi_\mathbb{B}|$ and $j = |U_\mathbb{B}| + j_\circ$, connect the pair of cap vertices $\theta^{2j-1}_\mathbb{B}$ and $\theta^{2j}_\mathbb{B}$ by a dummy adjacency edge, associating this pair to the dummy segment $\varphi^{j_\circ}_\mathbb{B} \in U^\varphi_\mathbb{B}$.

($4_\approx$) For each edge $AB \in \widehat{F}$, let $A \in \widehat{U}_\mathbb{A}$ and $B \in \widehat{U}_\mathbb{B}$ be associated, respectively, to the cap vertices $\theta^{2i-1}_\mathbb{A}, \theta^{2i}_\mathbb{A} \in \hat{\theta}(\mathbb{A})$ and $\theta^{2j-1}_\mathbb{B}, \theta^{2j}_\mathbb{B} \in \hat{\theta}(\mathbb{B})$. Connect the cap vertices with cap edges in the two crosswise possibilities: $\{\theta^{2i-1}_\mathbb{A}, \theta^{2j-1}_\mathbb{B}\}$, $\{\theta^{2i}_\mathbb{A}, \theta^{2j}_\mathbb{B}\}$ (1st of A to 1st of B, 2nd of A to 2nd of B), and also $\{\theta^{2i-1}_\mathbb{A}, \theta^{2j}_\mathbb{B}\}$, $\{\theta^{2i}_\mathbb{A}, \theta^{2j-1}_\mathbb{B}\}$ (1st of A to 2nd of B, 2nd of A to 1st of B). [Simple optimization: if the edge AB is a complete component in $\widehat{\mathcal{C}}(\mathbb{A},\mathbb{B})$ (that is, both vertices A and B have degree one) and, moreover, either A or B is a dummy segment, then, since both ends of a dummy segment are "equivalent", we simply remove one of the two crosswise connections of cap vertices described above (arbitrarily chosen).] The set of all cap edges is denoted by E_θ.

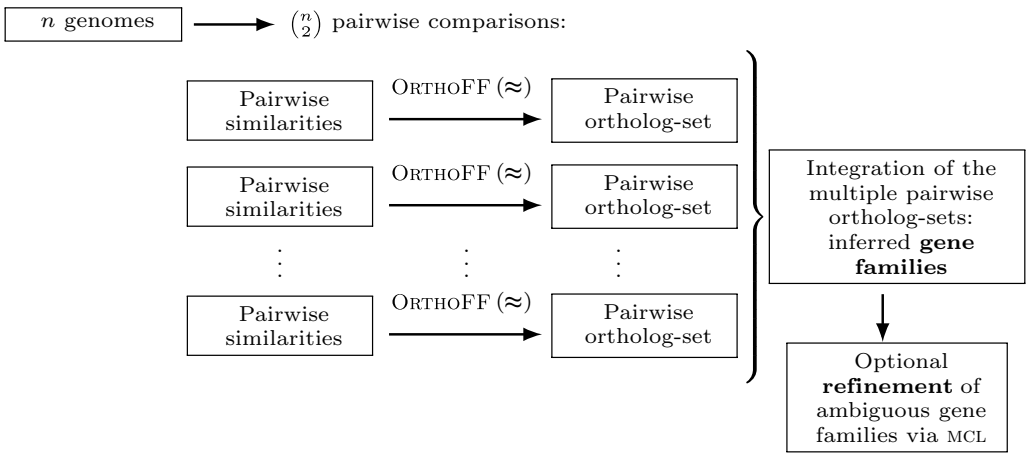


Fig. 5 The pipeline of our approach is straightforward: our gene families are the connected components of the *n*-partite graph derived by the integration of the computed ortholog-sets. The resulting ambiguous families can be optionally refined with the help of MCL

Denote by $\mathfrak{P}_{\approx}$ the set of all possible capping-sets (perfect matchings) between the vertices of $\hat{\theta}(\mathbb{A})$ and $\hat{\theta}(\mathbb{B})$. The optimization problem over $\theta_{\approx}(\mathit{FFR}(\mathbb{A},\mathbb{B},\mathcal{S}))$ is defined as

$$\textsc{GenDiFF}{\approx}(\mathbb{A},\mathbb{B},\mathcal{S}) = \min_{\mathcal{L}\in\mathfrak{S},P\in\mathfrak{P}_{\approx}} \{\mathrm{wd}^{\mathrm{ID}}_{\mathrm{DCJ}}(\widehat{\mathrm{D}}[\mathcal{L},\mathrm{P}])\}\,.$$

Assuming that a sibling set $\mathcal{L}_{\approx}$ (together with one of its best heuristic capping-sets) gives the optimal solution for $\textsc{GenDiFF}{\approx}(\mathbb{A},\mathbb{B},\mathcal{S})$, a best heuristic ortholog-set is $\textsc{OrthoFF}{\approx}(\mathbb{A},\mathbb{B},\mathcal{S}) = \mathcal{O}_{\mathcal{L}_{\approx}}$. Both problems $\textsc{GenDiFF}{\approx}$ and $\textsc{OrthoFF}{\approx}$ can also be solved with the ILP FF-DCJ-Indel, shown in Additional file 1: Section (A).

Search space compared to optimal capping

If the threshold τ is similar to the numbers $\kappa(\mathbb{A})$ and $\kappa(\mathbb{B})$ of linear segments in each genome, and in the unlike situation where all linear segments from $\mathbb{A}$ are connected to all linear segments from $\mathbb{B}$ in $\widehat{\mathcal{C}}$, the heuristic capping induced by $\widehat{\mathcal{C}}$ may be as large as the optimal capping.

Therefore, τ is thought to be smaller than $\kappa(\mathbb{A})$ and $\kappa(\mathbb{B})$, effectively reducing the number of capping-sets. We could not yet estimate this reduction as a function of τ, though. As our experimental results with real genomes show (details below, in the next section), with a small τ the heuristic capping leans to a considerably smaller number of capping-sets in practice.

Integration of pairwise heuristic ortholog-sets into gene families

The ILP FF-DCJ-Indel solving $\textsc{OrthoFF}{\approx}$ (with heuristic capping) is the core of a new version of our tool, called $\textsc{OrthoFFGC}{\approx}$, for inferring gene families across several species, as illustrated in Fig. 5. Recall that each family is a connected component of the *n*-partite graph obtained by the simple integration of the computed pairwise ortholog-sets. An *ambiguous* family corresponds to a connected component of the *n*-partite graph that has more than one gene from the same genome. Otherwise we have a *resolved* family, which can be either *complete*, when it contains one gene per genome, or *incomplete* otherwise. Figure 6 illustrates these types of families in a 3-partite graph.

Refinement of ambiguous gene families via MCL

Both pipelines OrthoFFGC and $\textsc{OrthoFFGC}{\approx}$ may produce a small number of large ambiguous families with low connectivity that can benefit from a further refinement (e.g. by removing bridges or making edge cuts in

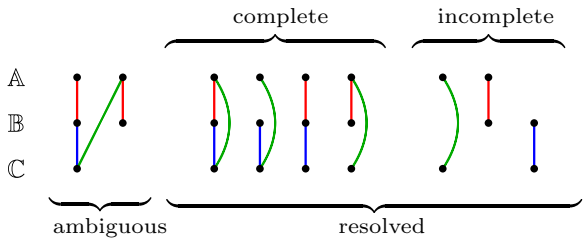


Fig. 6 Types of families given by the integration of three ortholog-sets into a 3-partite graph

areas with small edge-connectivity) in order to produce more cohesive families. In an optional extra step, as shown in Fig. 5, our pipeline refines ambiguous families with the help of MCL [19], a fast and scalable clustering algorithm based on simulation of stochastic flow in graphs.

Implementation and experiments

The pipeline OrthoFFGC (with optimal capping) was previously integrated into the FFGC workflow [12, 22], which includes the pre-computation of gene similarities, allowing therefore the automatic generation of families directly from the genome data. We implemented the new pipeline OrthoFFGC≈ (with heuristic capping) as another extension of the same workflow. The optional MCL step for refining ambiguous families is integrated to both OrthoFFGC and OrthoFFGC≈. The implementation and its documentation can be downloaded from our GitLab server at https://gitlab.ub.uni-bielefeld.de/gi/FFGC or as a Conda package at https://anaconda.org/bioconda/ffgc. (Note that MCL is an external dependency, automatically installed only if the download is obtained from Conda.)

An important recent modification of our pipeline is that now it performs pairwise similarity computations by default via DIAMOND [23] and no longer via BLAST [24]. This change and some further optimizations improved greatly the running times of OrthoFFGC≈, that are now closer to the fastest alternative tools. Suprisingly, adopting DIAMOND also produced a small improvement of the quality of our gene families, compared to the version previously published [18]. The current values for quality and running times can be found in the end of this section.

Alternative inference tools used for comparison purposes

ProteinOrtho and Poff. ProteinOrtho [16] is a fast tool that clusters genes to find significant orthologous groups, based on a heuristic of reciprocal best hits of the corresponding sequence similarities. Its Poff extension [17] takes into account gene adjacencies as an additional criterion for the discrimination of orthologs.

Oma. Based on sequence similarities and on phylogeny, Oma [25] is the underlying tool of the homonym online orthology browser. The standalone tool allows custom genomes to be compared to infer orthologous groups. Two types of families are reported: *hierarchical orthologous groups* (OmaHOGs), which may include ambiguous families, and *resolved groups* (OmaGroups).

Gene similarities in OrthoFFGC/OrthoFFGC≈

The computation of gene similarities is done via FFGC and is described as follows. For any given pair of genes x and y, the software DIAMOND is used for computing the value $\textsc{bitscr}(x \to y)$, that corresponds to the bitscore of gene x *with respect to* gene y. If gene x is in genome $\mathbb{A}$ and gene y is in genome $\mathbb{B}$, both $\textsc{bitscr}(x \to y)$ and $\textsc{bitscr}(y \to x)$ must be taken into consideration for calculating the similarity $\sigma(x, y)$, that is equivalent to their *relative reciprocal score* [26]:

$$\sigma(x,y) = \frac{\textsc{bitscr}(x \to y) + \textsc{bitscr}(y \to x)}{\textsc{bitscr}(x \to x) + \textsc{bitscr}(y \to y)}.$$

Negligible similarities are then identified and discarded by the filter F, by requiring that (1) for a first given threshold $F_\epsilon \in (0, 1]$, $\sigma(x, y) \geq F_\epsilon$ (*absolute filter*); and (2) for a second given threshold $F_t \in (0, 1]$, the value $\textsc{bitscr}(x \to y)$ must be at least a F_t-fraction of the highest bitscore of x with respect to any gene in $\mathbb{B}$ and the value $\textsc{bitscr}(y \to x)$ must be at least a F_t-fraction of the highest bitscore of y with respect to any gene in $\mathbb{A}$ (*relative minimum reciprocal similarity for additional hits*, which was developed and used in the alternative tools ProteinOrtho [16] and Poff [17]). Once these conditions are fulfilled, the edge xy is added to the gene similarity graph with score $\sigma(xy) = \sigma(x, y)$. The adopted values are $F_\epsilon = 0.1$ and $F_t = 0.8$, the latter also being adopted for ProteinOrtho and Poff whenever these tools were used in our experiments.

The default values of the other parameters for the pre-computation of gene similarities via FFGC were kept, except for one of them: the minimum number of genomes for which each gene must share some similarity in is set to 1, otherwise genes not similar to any other gene, which should be still considered in indels, would not appear in the chromosomal gene order.

Computational environment and additional parameters of OrthoFFGC≈

We ran experiments in a 2.7GHz multi-core machine. Whenever possible, tasks ran using 8 cores. As an ILP solver, we used Gurobi.

For the post-processing refinement of ambiguous families in OrthoFFGC≈, we used the default parameters of MCL with a conservative "inflation" value of 1.4 suggested in its manual (https://micans.org/mcl).

In the following we will describe our experiments, based on genome assemblies fetched from NCBI. We performed two distinct comparisons. First, based on a set of

Table 1 ILP running times, DCJ-indel distances and weighted DCJ-indel distances with optimal and heuristic cappings for the pairwise comparisons of primate genomes

	Optimal capping						Heuristic capping ($\tau = 2$, $\epsilon = 0.1$)	
pair	$\lvert\mathcal{O}\rvert$		d_{DCJ}^{ID}		wd_{DCJ}^{ID}		time (m) / gap	
bonobo × chimp	20173	+3	2664	+18	4342.99	+10.44	1440 / 0.96%	60 / 0.89%
bonobo × gorilla	19515	≡	3318	≡	5585.24	+0.07	1440 / 0.21%	60 / 0.13%
bonobo × human	18441	≡	2975	≡	4958.24	≡	103 / -	12 / -
bonobo × orang	18454	≡	3402	+1	5994.34	+0.98	297 / -	42 / -
chimp × gorilla	19569	≡	3277	+1	5500.19	+1.00	1440 / 0.19%	60 / 0.12%
chimp × human	18518	−1	2945	≡	4762.08	+1.91	37 / -	3 / -
chimp × orang	18521	≡	3349	≡	5874.02	≡	174 / -	19 / -
gorilla × human	18481	≡	3056	≡	5085.42	≡	1440 / 0.10%	16 / -
gorilla × orang	18584	−1	3449	−2	6082.19	+0.06	317 / -	30 / -
human × orang	17858	≡	2875	+4	5067.17	+3.99	43 / -	4 / -

Table 2 Numbers of classified genes and families inferred by OrthoFFGC≈ and ProteinOrtho for the dataset of five primates

Method	**# of classified genes** (out of 107317)	**# of families**		
		ambiguous	**resolved incomplete**	**resolved complete**
OrthoFFGC≈	100872	656	3719	16726
intersection	98156	154	2016	14770
ProteinOrtho	100481	1795	2939	14810

five completely assembled primate genomes, we compared OrthoFFGC≈ families (inferred with the heuristic capping) to OrthoFFGC families (inferred with the optimal capping). Then, based on the set of 11 *Drosophilas* including partially assembled genomes, we compared OrthoFFGC≈ families to the gene families inferred by other tools, using FlyBase families as reference.

Analysis of completely assembled primate genomes: comparing OrthoFFGC≈ to OrthoFFGC

The goal of this experiment is to evaluate the impact of the heuristic capping on running times and on the quality of results, with respect to the optimal capping in a dataset of large genomes. We compared families inferred by OrthoFFGC and OrthoFFGC≈, considering the following five primate genomes: bonobo (*P. paniscus*), chimpanzee (*P. troglodytes*), gorilla (*G. gorilla*), human (*H. sapiens*) and orangutan (*P. abelii*). Those genomes comprise roughly 20,000 ∼ 22,000 genes distributed in 25 chromosomes, except for the human genome that has 24 chromosomes. The average number of edges (similarities) for each vertex (gene) in the gene similarity graphs is 1.09, totaling 429268 vertices and 234297 edges. Considering only the genes with multiple similarities, the average degree is 2.93.

For the capping heuristic in OrthoFFGC≈, we set $\epsilon = 0.1$ and $\tau = 2$, a choice that reduces significantly the number of capping sets, but still gives some options to the ILP solver. With these choices, the largest number of edges in $\widehat{\mathcal{C}}(\mathbb{A}, \mathbb{B})$ was for the comparison of gorilla and human. In this case, $\widehat{\mathcal{C}}$ has 27 edges distributed among 21 components with 2 vertices, and 2 components with 4 vertices (including 1 dummy segment). That corresponds to at most $\sim 1.1 \times 10^6$ capping-sets in $\theta_{\approx}$, while the pairwise comparison with the optimal $\theta_\star$ has 50! capping-sets.

This significant reduction of the search space enabled 7 out of 10 pairwise comparisons with the heuristic capping to finish within the time limit of 60 min, in contrast to the lengthy comparisons with the optimal capping, that were limited to 1440 min (24 h). The results are shown in Table 1, where we can also see that the size of the ortholog-sets $\mathcal{O}$, the number of DCJ-indel operations d_{DCJ}^{ID}, and the weighted rearrangement distances wd_{DCJ}^{ID} of the two approaches are almost identical for all pairwise comparisons. While 21090 families were inferred using OrthoFFGC, 21101 were inferred using OrthoFFGC≈, and 99.2% of those

Table 3 Numbers of classified genes and families inferred by the different methods for the dataset of 11 Drosophilas

Method	# of classified genes (out of 151493)	# of families		
		ambiguous	res. incomplete	res. complete
FlyBase	136190	1558	4769	6189
OmaHOGs	135406	1077	7213	5028
(∩FlyBase)	(124548)	(564)	(3151)	(4884)
OmaGroups	131523	-	9673	4688
(∩FlyBase)	(121775)	(-)	(2868)	(4323)
ProteinOrtho	136623	1151	6133	5569
(∩FlyBase)	(125638)	(610)	(3541)	(5373)
Poff	135364	439	7388	5758
(∩FlyBase)	(124884)	(226)	(3601)	(5391)
OrthoFFGC≈	130870	902	5813	5899
(∩FlyBase)	(122048)	(414)	(3406)	(5658)
OrthoFFGC≈ + mcl	130870	648	6458	6072
(∩FlyBase)	(122048)	(374)	(3598)	(5687)

families are the same, showing that moving from the optimal capping to the heuristic capping of the family-free relational diagram had very little impact on the gene family inference. The total running time of OrthoFFGC≈ was of about 321 min (of which 306 min were spent by the pairwise ILP computations). The numbers of classified genes and of inferred families are displayed in Table 2.

This experiment accomplished its goal, by showing that the heuristic capping can have the same quality of the optimal capping, while having substantial impact in speeding up the ILP computations. Indeed, the time required to analyze the dataset of five primates with OrthoFFGC≈ was quite reasonable. However it was still considerably longer than the fastest alternative tool: the total running time of ProteinOrtho for the same dataset was of about 19 min only, with the corresponding numbers of classified genes and inferred families being also displayed in Table 2. It is worth mentioning that in OrthoFFGC≈ a resolved family has the additional confidence of resulting from independently computed pairwise ortholog-sets. Therefore it is remarkable that OrthoFFGC≈ could identify a significantly higher number of resolved complete families, including almost all resolved complete families also inferred by ProteinOrtho.

Analysis of partially assembled *Drosophila* (fruit flies) genomes

The FlyBase consortium (https://flybase.org) sequenced, assembled and annotated the genomes of 12 *Drosophilas* with ∼ 12,000–16,000 protein-coding genes (for genes with multiple transcripts, we kept only the longest), however only 11 of those genomes are available on NCBI together with the complete annotation: *D. ananassae*, *D. erecta*, *D. grimshawi*, *D. melanogaster*, *D. mojavensis*, *D. persimilis*, *D. sechellia*, *D. simulans*, *D. virilis*, *D. willistoni* and *D. yakuba*.

In the family-free analysis, the total number of vertices and edges in the gene similarity graphs are 1514930 and 633521, respectively, with average degrees of 0.83 considering all vertices, and 2.22 looking only at those genes with multiple similarities. For the capping heuristic in OrthoFFGC≈, we again set $\epsilon = 0.1$ and $\tau = 2$.

Average numbers of cap edges and capping-sets in $\theta_\star$ and in $\theta_\approx$

The analyzed *Drosophila* genomes have 507 contigs on average, therefore each optimally capped family-free relational diagram has $1{,}014 \times 1{,}014 = 1{,}028{,}196$ cap edges and an unfeasible total of 1, 014! capping-sets on average.

In contrast, considering the perfect shared-content graphs for all pairwise *Drosophila* comparisons, 99.7% of the components in those graphs have only 1 linear segment in each part of the graph. In the remaining 0.3%, 80% have 7 or fewer linear segments in each part, with the largest component having 76 linear segments in each part. The perfect shared-content graphs have an average of 1,419 edges. For that number of edges, each heuristically capped family-free relational diagram has 5,676 cap edges on average. As the exact number of perfect matchings in arbitrary graphs is not trivial to estimate, we computed an upper limit for the average number of distinct capping-sets by the Bregman-Cinc inequality [27] of the permanent of a squared matrix: ∼ 45!, with median ∼ 18!.

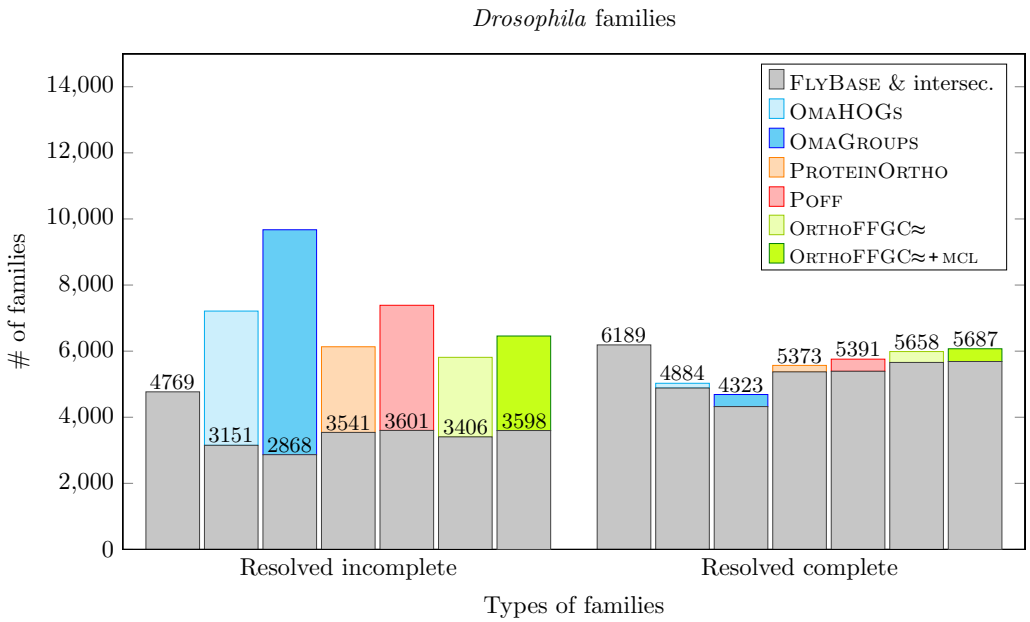


Fig. 7 The numbers of resolved incomplete and complete families in FlyBase, followed by the numbers of families inferred by OmaHOGs, OmaGroups, ProteinOrtho, Poff and OrthoFFGC≈ (without and with mcl refinement). The lower part of each bar represents the intersection between the inferred sets and FlyBase. (For resolved complete families, the numbers of families in the intersections are shown on the top of bars)

Method	TP	FP	FN	precision	recall	F_1-score
OmaHOGs	553,507	22,650	61,336	0.961	0.900	0.929
OmaGroups	500,500	1,618	114,343	0.997	0.814	0.896
ProteinOrtho	569,600	22,983	45,243	0.961	0.926	0.943
Poff	537,469	4,062	77,374	0.992	0.874	0.930
OrthoFFGC≈	559,220	31,150	55,623	0.947	0.910	0.928
OrthoFFGC≈ + mcl	547,931	4,126	66,912	0.993	0.891	0.939

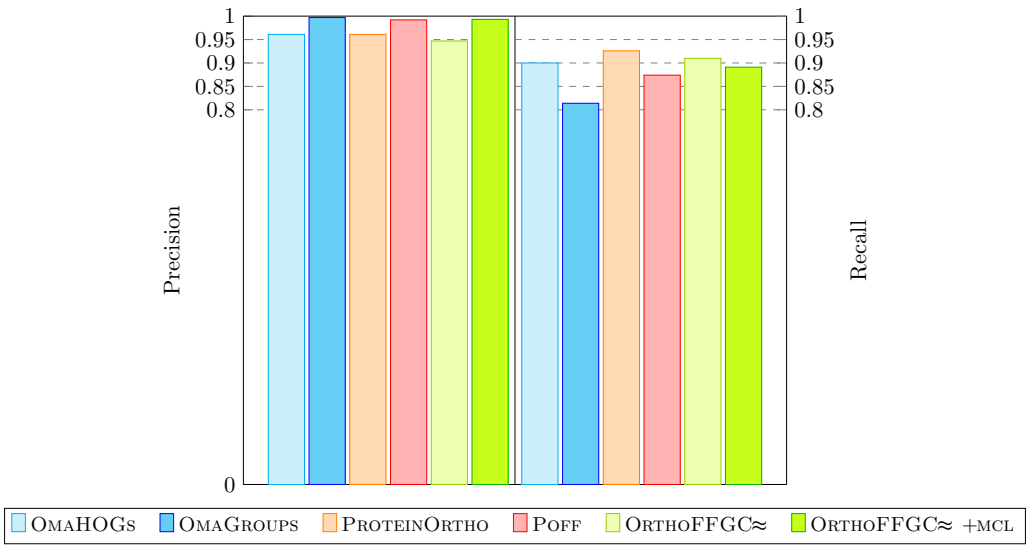


Fig. 8 Precision $\left(\frac{TP}{TP+FP}\right)$, recall $\left(\frac{TP}{TP+FN}\right)$ and their harmonic mean F_1-score for OmaHOGs, OmaGroups, ProteinOrtho, Poff and OrthoFFGC≈ (without and with mcl refinement), based on the dataset with eleven *Drosophilas*

Table 4 Running times for computing *Drosophila* families

	Step	OMA	PROTEINORTHO	POFF	ORTHOFFGC$\widetilde{\widetilde{H}}$
1.	Sequence alignments and similarities	200h	34m$^{\diamond}$	34m$^{\diamond}$	13m$^{\diamond}$
2.	Addit. orthology criterion (synteny/rearrangements)	-	-	9m (adjacencies)	8m (ILPs)
3.	Build / finish building families (and refine*)	1h	15s	26s	1m (+2m*)
	Total	**201h**	**34m**	**43m**	**22m (+2m*)**

$^{\diamond}$ Computed with DIAMOND

* The optional additional MCL refinement step on ORTHOFFGC≈ takes 2 extra minutes

Benchmark for our experiments

Reference families were obtained directly from FLYBASE (https://flybase.org). Since the set of genes classified in FLYBASE is slightly different from the set of genes present with their coding sequences in database files, we filtered out a small portion ($\sim 7\%$) of genes in FLYBASE families so that only those present in the NCBI databases with their coding sequences were kept. Prior to any comparison of inferred families to FLYBASE families, we also filtered out from the inferred families genes not present in FLYBASE families.

***Comparing* ORTHOFFGC≈ *to* PROTEINORTHO, POFF *and* OMA**

We analyzed the dataset of 11 *Drosophila* genomes for comparing the performance of ORTHOFFGC≈ against PROTEINORTHO, POFF and OMA, providing as additional input to the latter the phylogenetic tree (obtained from FLYBASE) of the same 11 *Drosophilas*. Unlocalized contigs were not filtered out, resulting in genomes with 11 to 1041 linear segments.

Quality of inferred families The numbers of families in FLYBASE and those inferred by OMAHOGS, OMAGROUPS, PROTEINORTHO, POFF and ORTHOFFGC≈ are all bigger than 12,000. In order to have a hint on the quality of results, we focus on the intersections with FLYBASE families and on *precision* and *recall* values computed for all methods.

The numbers of classified genes and families inferred by all methods, together with the respective intersections with FLYBASE are shown in Table 3. Figure 7 shows the comparative picture focusing on the numbers of resolved incomplete and complete families for all methods.

We also counted the numbers of pairwise gene homologies that are classified as *true positive* (TP), *false positive* (FP) and *false negative* (FN) as follows. First, denote a subset of size two by *2-subset*. Now let $\mathcal{H}_{\text{FLY}}$ be the set composed of 2-subsets of all FLYBASE families, and, for any considered set of families X, let $\mathcal{H}_X$ be the set composed of 2-subsets of all families in X. Then TP of X is the size of $\mathcal{H}_{\text{FLY}} \cap \mathcal{H}_X$, FN of X is the size of $\mathcal{H}_{\text{FLY}} \setminus \mathcal{H}_X$ and FP of X is the size of $\mathcal{H}_X \setminus \mathcal{H}_{\text{FLY}}$. Based on that we computed the values of *precision* $\left(\frac{\text{TP}}{\text{TP+FP}}\right)$ and *recall* $\left(\frac{\text{TP}}{\text{TP+FN}}\right)$ for the sets of families inferred by the considered methods.

The results (Fig. 8) show that, while all methods performed quite well, our tool ORTHOFFGC≈ had the lowest precision. However, when the optional MCL refinement step is enabled in our pipeline, its overall results were improved, with an increase of the precision that is bigger than the corresponding decrease of the recall, reflected on the increase of the F_1-score.

Running times We compiled the running times of the four methods in Table 4. The (preprocessing) step 1 of computing the pairwise sequence similarities is required by all methods, but, while OMA has an internal implementation of the Smith-Waterman algorithm, the other methods used DIAMOND [23], which is the fastest known tool for accomplishing that task.

Having at hand the sequence similarities, PROTEINORTHO and OMA build families taking into consideration alignments and similarities. Additionally, OMA takes into consideration the provided phylogenetic tree of the 11 *Drosophilas*. The running times of these procedures are given in step 3. For the other methods, we separated in step 2 core procedures that use additional criteria to find pairwise orthologs: POFF does it via the analysis of synteny by means of conserved gene adjacencies, while ORTHOFFGC≈ generates and solves the ILPs for obtaining pairwise ORTHOFF≈ gene orthologies.

Adopting DIAMOND for similarity computations instead of BLAST was an important recent modification of our pipeline. This change and the use of more restrictive parameters for the heuristic capping improved greatly the running times of our method compared to its previous version [18]. With these improvements the running times of OrthoFFGC≈ are now the fastest among the compared tools for the *Drosophila* dataset.

Conclusions and discussion

We devised and implemented a heuristic capping for improving our recently developed pipeline OrthoFFGC [13] for inferring gene families based on genome rearrangements. In OrthoFFGC we adopted an optimal capping including all connections between the ends of linear segments to allow all possible $(2p_*)!$ capping-sets in the input of the ILP FF-DCJ-Indel that infers the OrthoFF pairwise orthologs. However, due to the heavy optimal capping, FF-DCJ-Indel can hardly converge when analyzing larger genomes (such as mammals) and fails in handling a pair of genomes if one or both of them (with at least the dimension of a fruit fly genome) are distributed in a hundred contigs.

In contrast, the new pipeline OrthoFFGC≈ adopts a lighter heuristic capping including connections only between linear segments that share gene content. This leads to a much smaller number of capping-sets in the input of the same FF-DCJ-Indel that here infers OrthoFF≈ pairwise orthologs. Despite the use of a heuristic capping, our evaluation showed that the quality of the orthologies inferred by OrthoFFGC≈ was very good.

A first evaluation on a dataset of five completely assembled primate genomes was done by comparing the families inferred by the previous workflow OrthoFFGC with the new OrthoFFGC≈. The results showed that the gene families inferred by the two pipelines are virtually the same. Therefore, in practice, the heuristic capping did not have a negative impact on the inferred gene families, essentially preserving the original (optimal) orthology relations.

A second evaluation was done on a dataset of 11 *Drosophila* genomes, including partially assembled genomes distributed in several contigs, by adopting the gene families curated by the FlyBase consortium as a benchmark. In this experiment we compared OrthoFFGC≈ to other genome-scale methods, namely Oma, ProteinOrtho and Poff. The running times of OrthoFFGC≈ for the *Drosophila* dataset are better than the fastest alternative tools ProteinOrtho and Poff, showing that our implementation is very efficient, despite the pairwise ILP computations. Furthermore, our results showed that OrthoFFGC≈ was able to infer only 3 resolved families less than ProteinOrtho and had the highest number of resolved complete families in common with FlyBase, and these intersections were improved after the refinement of ambiguous families with MCL. Concerning the analysis of pairwise gene orthologies derived from the inferred families, all tools had a very good performance. After the MCL refinement our tool reached the second best F_1-score, with a difference of only 0.004 to the best F_1-score achieved by ProteinOrtho.

The bottleneck of our pipeline is still the ILP pairwise computations that, despite the gain of heuristic capping, solve instances of an NP-hard problem. However, the heuristic capping allows to efficiently analyze large genomes such as mammals and, at least for genomes with the dimension of a fruit fly genome, OrthoFFGC≈ lifts the limitation of requiring chromosome-level assembled genomes, expanding to a great extent its applicability.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s13015-023-00238-y.

Additional file 1. Online supplemental material

Author contributions

DPR and MDVB designed the model. DPR implemented the heuristic capping and carried out the experiments. DPR and MDVB wrote the manuscript, whose final version was reviewed by both of them.

Funding

Open Access funding enabled and organized by Projekt DEAL.

Availibility of data and materials

See https://gitlab.ub.uni-bielefeld.de/gi/FFGC or https://anaconda.org/bioconda/ffgc.

Declarations

Ethics approval and consent to participate

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Received: 20 December 2022 Accepted: 17 August 2023
Published online: 28 September 2023

References

1. Bergeron A, Mixtacki J, Stoye J. A unifying view of genome rearrangements. In: Proc. of WABI. Lecture Notes in Bioinformatics, 2006;4175:163–173.
2. Hannenhalli S, Pevzner PA. Transforming men into mice (polynomial algorithm for genomic distance problem). In: Proc. of FOCS, 1995:581–592.
3. Braga MDV, Willing E, Stoye J. Double cut and join with insertions and deletions. J Comput Biol. 2011;18(9):1167–84.
4. Sankoff D. Genome rearrangement with gene families. Bioinformatics. 1999;15(11):909–17.
5. Bryant D. The complexity of calculating exemplar distances. In: Sankoff D, Nadeau JH, editors. Comparative Genomics. Computational Biology Series, vol. 1. London: Kluver Academic Publishers; 2000. p. 207–11.
6. Angibaud S, Fertin G, Rusu I, Thévenin A, Vialette S. On the approximability of comparing genomes with duplicates. J Graph Algo App. 2009;13(1):19–53.
7. Shao M, Lin Y, Moret B. An exact algorithm to compute the double-cut-and-join distance for genomes with duplicate genes. J Comput Biol. 2015;22(5):425–35.
8. Bohnenkämper L, Braga MDV, Doerr D, Stoye J. Computing the rearrangement distance of natural genomes. J Comput Biol. 2021;28(4):410–31.
9. Yancopoulos S, Attie O, Friedberg R. Efficient sorting of genomic permutations by translocation, inversion and block interchange. Bioinformatics. 2005;21(16):3340–6.
10. Braga MDV, Chauve C, Doerr D, Jahn K, Stoye J, Thévenin A, Wittler R. The potential of family-free genome comparison. In: Chauve C, El-Mabrouk N, Tannier E, editors. Models and Algorithms for Genome Evolution, vol. 19. Computational Biology Series. Berlin: Springer; 2013. p. 287–307.
11. Martinez FV, Feijao P, Braga MDV, Stoye J. On the family-free DCJ distance and similarity. Algorithms Mol Biol. 2015;13(10):777–80.
12. Rubert DP, Martinez FV, Braga MDV. Natural Family-Free Genomic Distance. Algorithms Mol Biol. 2021;16(4):1–6.
13. Rubert DP, Doerr D, Braga MDV. The potential of family-free rearrangements towards gene orthology inference. J Bioinform Comput Biol. 2021;19(6):2140014.
14. Dessimoz C, Cannarozzi G, Gil M, Margadant D, Roth ACJ, Schneider A, Gonnet GH. OMA, a comprehensive, automated project for the identification of orthologs from complete genome data: introduction and first achievements. In: Proc. of RECOMB-CG. Lecture Notes in Bioinformatics, 2005;3678:61–72.
15. Roth ACJ, Gonnet GH, Dessimoz C. Algorithm of OMA for large-scale orthology inference. BMC Bioinform. 2008;9(518):1.
16. Lechner M, Findeiß S, Steiner L, Marz M, Stadler PF, Prohaska SJ. Proteinortho: Detection of (co-)orthologs in large-scale analysis. BMC Bioinform. 2011;12(124):1–9.
17. Lechner M, Hernandez-Rosales M, Doerr D, Wieseke N, Thévenin A, Stoye J, Hartmann RK, Prohaska SJ, Stadler PF. Orthology detection combining clustering and synteny for very large datasets. PLoS ONE. 2014;9(8):e105015.
18. Rubert DP, Braga MDV. Gene Orthology Inference via Large-Scale Rearrangements for Partially Assembled Genomes. In: Proc. of WABI. Leibniz International Proceedings in Informatics (LIPIcs), 2022;242 (24):1–22.
19. van Dongen S. Graph clustering via a discrete uncoupling process. SIAM Journal on Matrix Analysis and Applications. 2008;30(1):121–41.
20. Hall P. On representatives of subsets. J London Mat Soc. 1935;s1–10(1):26–30.
21. Tassa T. Finding all maximally-matchable edges in a bipartite graph. Theoret Comput Sci. 2012;423:50–8.
22. Doerr D, Feijão P, Stoye J. Family-free genome comparison. In: Setubal JC, Stoye J, Stadler PF, editors. Comparative Genomics: Methods and Protocols. Methods in Molecular Biology, vol. 1704. New York: Springer; 2018. p. 331–42.
23. Buchfink B, Xie C, Huson DH. Fast and sensitive protein alignment using DIAMOND. Nat Methods. 2015;12:59–60.
24. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. J Mol Biol. 1990;215(3):403–10.
25. Altenhoff AM, Levy J, Zarowiecki M, Tomiczek B, Vesztrocy AW, Dalquen DA, Müller S, Telford MJ, Glover NM, Dylus D, et al. OMA standalone: orthology inference among public and custom genomes and transcriptomes. Genome Res. 2019;29(7):1152–63.
26. Pesquita C, Faria D, Bastos H, Ferreira AE, Falcão AO, Couto FM. Metrics for GO based protein semantic similarity: a systematic evaluation. BMC Bioinformat. 2008;9(Suppl 5):4.
27. Friedland S. An upper bound for the number of perfect matchings in graphs; 2008. arXiv:0803.0864.

Publisher's Note