BMC Bioinformatics

SOFTWARE

Open Access

VIRI: a visualization tool for tree reconciliations

Maurizio Patrignani[1], Giordano Dionisi[2], Blerina Sinaimeri[3*] and Tiziana Calamoneri[2*]

*Correspondence:
Blerina Sinaimeri
bsinaimeri@luiss.it
Tiziana Calamoneri
calamo@di.uniroma1.it
[1]Department of Civil, Computer and Aeronautical Engineering, Roma Tre University, Via della Vasca Navale 79, 00146 Rome, Italy
[2]Computer Science Department, Sapienza University of Rome, Viale Regina Elena 295, 00161 Rome, Italy
[3]Department of AI, Data and Decision Sciences, LUISS Guido Carli University, Viale Romania 32, 00197 Rome, Italy

Abstract

Background Cophylogeny reconciliation is a powerful method for analyzing host-symbiont coevolution. The cophylogeny problem consists of mapping the phylogenetic tree of the symbionts into the one of the hosts, including events such as duplications, co-speciation, host-switches, and extinctions by comparing the discrepancies between the topologies of the associated symbiont evolutionary trees. Visualizing tree reconciliations is important for biologists as it aids in understanding and identifying specific patterns in the coevolution of hosts and symbionts. Additionally, when multiple optimal solutions exist, it allows for the quick comparison of different reconciliations between the same pair of trees.

Results Here, we present VIRI (visual inspector of reconciliation instances), a new tree reconciliation visualizer. We adopt a hybrid metaphor combining space-filling (for host trees) and node-link (for symbiont trees) approaches, implementing the algorithms described in Calamoneri et al. (Theor Comput Sci 815:228–245. https://doi.org/10.1016/j.tcs.2019.12.024, 2020). The visualizations produced by VIRI are designed to be clear and interpretable, thanks to an unambiguous, top-down layout of tree reconciliations and the preservation of the user's mental map when comparing multiple reconciliations on the same pair of trees. In particular, the consistent use of a shared host tree layout across visualizations is a novel feature that facilitates direct comparison. Moreover, VIRI proposes a crossing-free visualization whenever possible. Finally, VIRI allows users to store datasets and download their visualizations, offering a convenient way to organize and share data. An example of visualization produced by VIRI is depicted in Fig. 1.

Conclusions VIRI efficiently produces clear and easy-to-read visualizations of tree reconciliations. VIRI is free and available at https://viri.di.uniroma1.it/.

Keywords Tree reconciliation, Space-filling approach, Visualization tool

Background

Cophylogeny reconciliation is a powerful method for analyzing host-symbiont coevolution (see e.g. [1–6]). It consists of mapping the phylogenetic tree of the symbionts into the one of the hosts. Such mapping, called a *reconciliation*, allows the identification of four types of biological events: (a) *cospeciation*, when the symbiont diverges in correspondence to the divergence of a host species; (b) *duplication*, when the symbiont diverges but not the host; (c) *host switch*, when a symbiont switches from one

host species to another independently of any host divergence; and (d) *loss*, which can describe, for instance, the speciation of the host species independently of the symbiont, which then follows just one of the new host species.

Visualizing cophylogeny reconciliations is essential for the researchers to get biological insights into the data. Moreover, for the same dataset, there are often several optimal solutions. Thus, it becomes necessary to be able to visualize them in order to easily compare different reconciliations between the same pair of trees by quickly looking at them and better understanding the differences.

The representation conventions adopted by the available tools for computing and exploring reconciliations can be roughly classified into two families, the first one represents the two trees next to each other, with the traditional node-link metaphor, where the nodes of the symbiont tree are drawn close to the nodes of the host tree they are associated with; the second one represents the host tree as a background shape, such that its nodes are shaded disks and its arcs are thick pipes, while the symbiont tree is drawn in the traditional node-link style, inside the pipes representing the host tree.

The advantage of the first strategy lies in its simplicity and in the availability of several algorithms for drawing binary trees in small areas. However, when several symbiont nodes are associated with the same host node, the drawing becomes cluttered, and the attribution of symbiont nodes to host nodes becomes unclear. Additionally, overlapping the host and symbiont trees often results in a high number of crossings.

The straight-line representation of the trees obtained with CoRe-PA [7] and orthogonal drawings output by Jane 4 [8] and by eMPRess [4] fall within this strategy (see Figs. 2a, b and 3a).

The tools in the second category are many including: CophyTrees (the viewer associated with Eucalypt [11]), Treerecs [12], DoubleRECVIZ [13] (see Figs. 3b, 4a, b).

Moreover, Thirdkind [15] (see Fig. 5a) is able to show both reconciliations (two levels) and pairs of reconciliations, the gene/symbiont and the symbiont/host ones, in a single representation (three levels).

All these representations are particularly effective, as they are unambiguous. However, they are still cluttered when a symbiont subtree has to be squeezed inside the reduced area of a host node. Moreover, in some cases (e.g., CophyTrees), a greedy strategy to reduce the crossings between the two trees is adopted but, in fact, it does not effectively reduce the crossings in the presence of multiple host switches.

Reconciliation viewers like Primetv [14] and SylvX [16] (see Figs. 5b and 6) adopt this strategy, using a model that maps the symbionts to both edges and nodes. This is effective when one wants to convey a degree of uncertainty in the association of symbionts to hosts, but is not appropriate for representing our model of reconciliation,

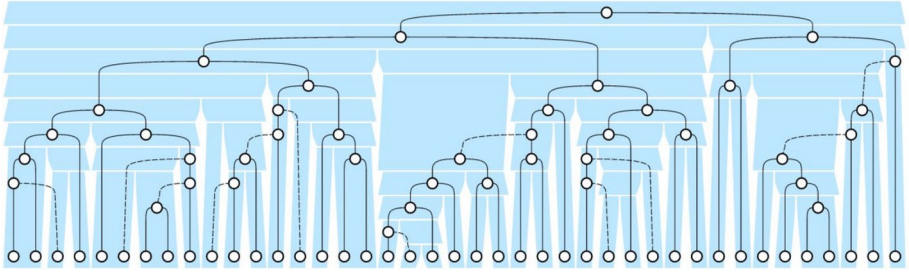
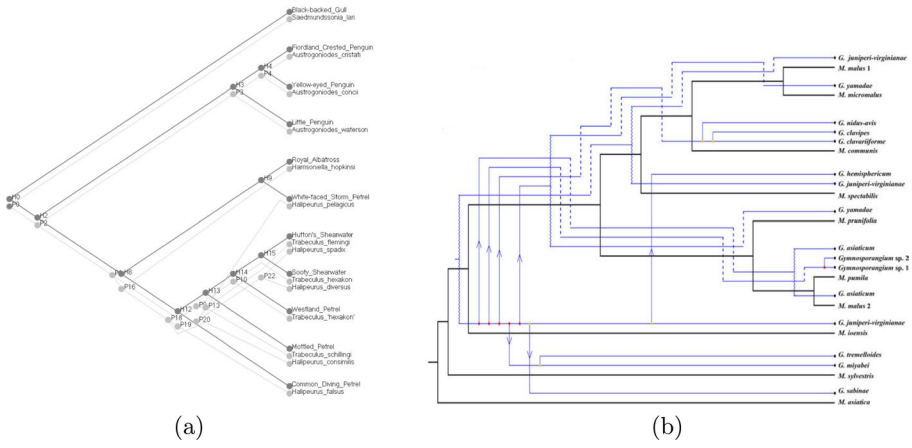
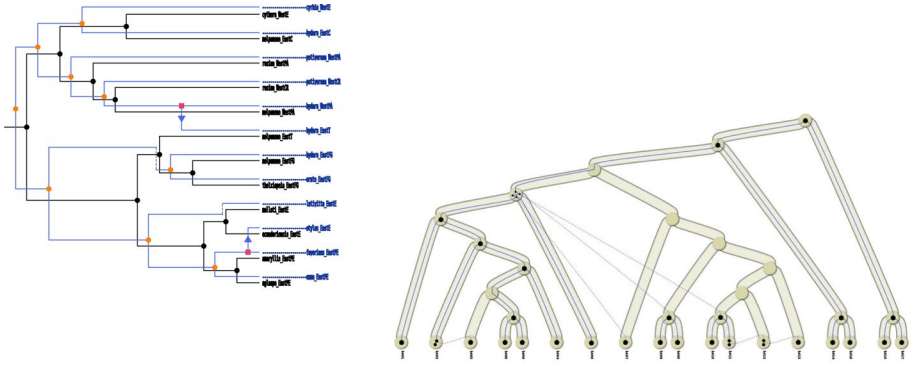


Fig. 1 An example of output of VIRI



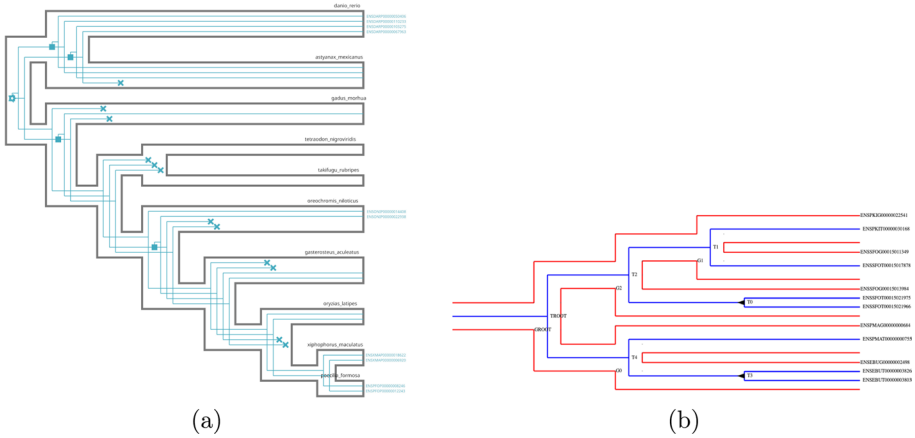
(a) (b)

Fig. 2 **a** Visualization of a reconciliation of *seabirds & chewing lice* coevolution trees obtained with CoRe-PA [7]. The image is taken from CoRe-PA tutorial [9]; **b** cophylogenetic analysis of the gymnosporangium-malus pathosystem conducted with Jane 4 [8]; the image is taken from [10]



(a) (b)

Fig. 3 **a** An example of cophylogenetic tree drawn by eMPRess; (courtesy of the authors of [4]); **b** reconciliation drawn by CophyTrees, the viewer associated with Eucalypt [11]



(a) (b)

Fig. 4 **a** An example of a cophylogenetic tree drawn by Treerecs (courtesy from the authors of [12]); **b** cophylogenetic tree produced running DoubleRECVIZ [13]

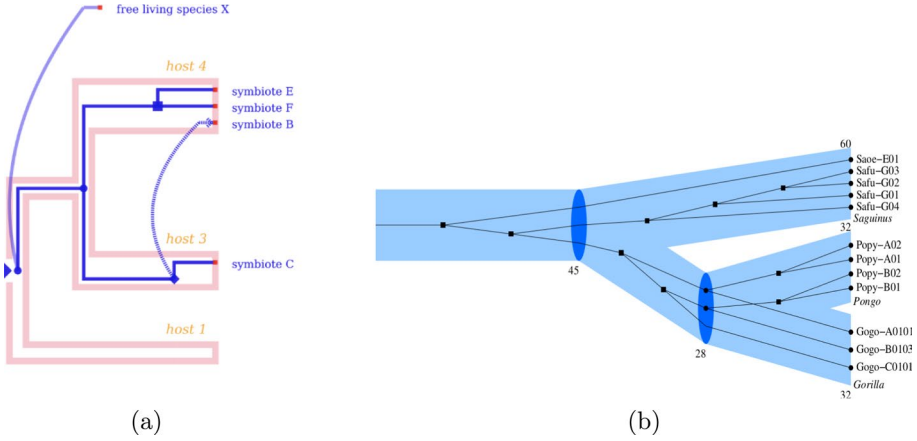


Fig. 5 **a** An example of a cophylogenetic tree drawn by TreeKind. The image is taken from https://github.com/simonpenel/thirdkind/wiki; **b** Primetv-generated illustration of the reconciled tree showing the evolution of the gene family major histocompatibility complex class I in gorilla, orangutan, and tamarin. The image is taken from [14]

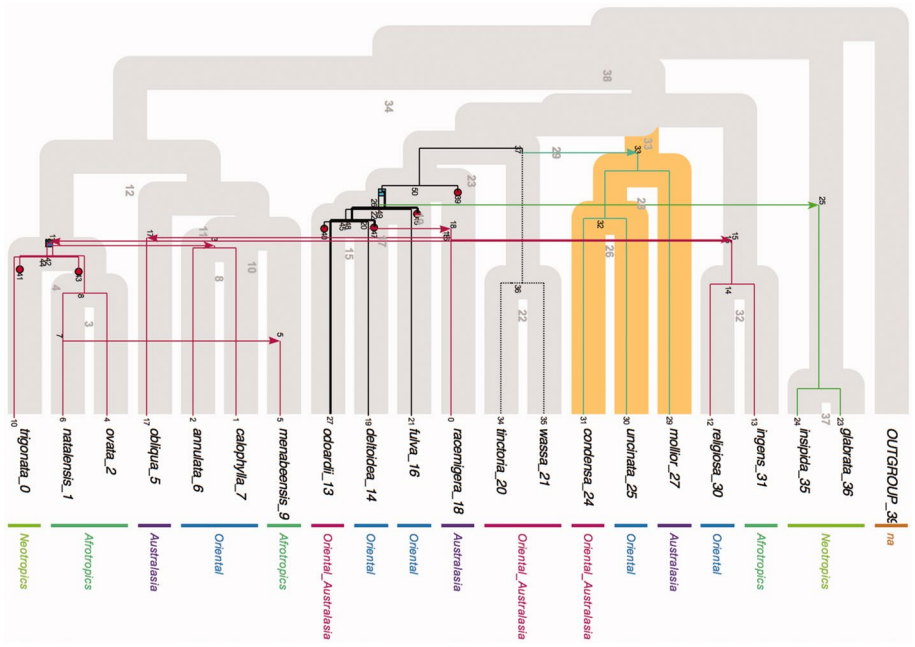


Fig. 6 An example of cophylogenetic tree drawn by SylvX. The image is taken from(http://www.sylvx.org/)

requiring a clear association of symbionts to hosts. Moreover, like the previously mentioned viewers, it also suffers from becoming cluttered when numerous host switches occur, which complicates even more the visualization and data interpretation.

In view of the weaknesses highlighted in each kind of visualization strategy, it is important to search for an alternative metaphor. In particular, space-filling methods, presenting extensive hierarchical or clustered data, seem to be attractive. Icicle [17] is a space-filling approach where the nodes are represented through a series of rectangles, and the arcs are represented by the contact of rectangles, such that the bottom side of the rectangle representing a node touches the top sides of the rectangles representing its children. The strength of space-filling visualizations, in general, lies in their ability to efficiently present large datasets in a two-dimensional space. Icicles have already been

applied successfully in bioinformatics, for example, in visualizing phylogenetic trees [18] or transcriptome data [19].

It should also be said that most of the surveyed tools are integrated within specific reconciliation applications and cannot visualize reconciliations produced by other algorithms. This means the visualization is typically the direct output of their reconciliation process, and it is not possible to bypass this step by transforming the output of another reconciliation tool into the input format required by the visualizer.

In this context, we propose VIRI (visual inspector of reconciliation instances), a new tree reconciliation visualizer, which is conceived to overcome the limitations of the previous approaches as detailed in the following sections.

Implementation

VIRI is a web application allowing users to visualize their own reconciliations as output by any tool reconciling pairs of trees (*e.g.*, Capybara [5]). By simply uploading a `.nex` file and the corresponding `.out` file, it is possible to visualize the corresponding reconciliations contained in the `.out` file. Notice that reconciliations generated by other tools can also be used, provided a simple conversion script is written to match the Capybara .out format. VIRI is designed with sharing in mind, so users can register and share their own datasets with the wider community, making it easier for others to build on their work and contribute to the field of cophylogenetics. Alternatively, they can just use VIRI to visualize their reconciliations without sharing them.

VIRI is an interactive tool. When hovering over a node with the mouse, the name of the associated species appears, and both the parent and children nodes are highlighted (see Fig. 7). Symbiont nodes are highlighted in yellow (parent) and red (children), while host nodes are highlighted in a darker shade of light blue compared to the other rectangles representing hosts.

Sometimes, the optimal solutions may include host-switches of the symbiont that result in a contradictory chronological ordering for the internal nodes of the host tree, resulting in a co-evolutionary history that is biologically inconsistent (time-unfeasible). In such cases, when the reconciliation is time-unfeasible (see [20, 21] for the formal definition of time-unfeasible reconcilation), we highlight the switch edges that result in time inconsistency by drawing them in orange (see, for example, Fig. 12b)

VIRI also includes a drop-down menu that allows users to choose various visualization options. These options include selecting straight or curved edges, deciding whether leaves are aligned to the floor level, setting the dimensions of the drawing, and specifying the space width between rectangles in the icicle plot.

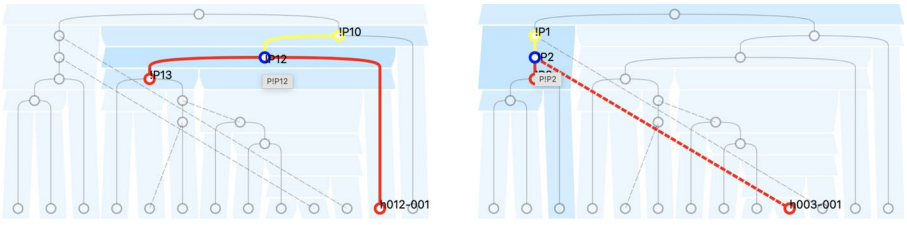


Fig. 7 Two screenshots of the output of VIRI for a reconciliation of *Rodents & Pinworms* [22], which highlight what can be seen when hovering over a portion of the drawing

Finally, the user can choose from the following visualization heuristics, each designed to reduce the number of crossings (indeed, minimizing crossings is an NP-hard problem [20]).

Heuristic `ShortenHostSwitches` (see as an example Fig. 8a) comes from [20] and is based on the observation that 'long' host-switch arcs are more likely to cause crossings than 'short' ones; hence, it searches for an embedding of H that reduces the distance between the end-nodes of host-switches of S; even the placement of symbiont nodes inside the representation of host nodes is done trying to avoid crossings.

Heuristic `SearchMaximalPlanar` (see as an example Fig. 8b) also comes from [20] and is based on the strategy of drawing first a large planar portion of the given reconciliation and then adding non-planar arcs. To this aim, we construct an auxiliary graph G, composed by the trees H and S, with their leaves joined by an edge if and only if there exists a symbiotic relation. We then construct a maximal planar subgraph of graph G using the Graph Drawing Toolkit GDT [23, 24]. If G is not planar, we iteratively remove edges that cause crossings until the remaining subgraph is planar. This subgraph is drawn in a planar fashion (that is, crossing-free), and the previously removed edges are reintroduced in a post-processing step.

The heuristic `RandomMethod` (illustrated in Fig. 8c) embeds the host tree by randomly selecting which child will be placed on the left for each internal node. This method serves as a pre-processing step for the subsequent heuristic, `HierarchySorting`, and is presented here both for its inherent interest and because it offers a baseline for comparison with the other methods.

Heuristic `HierarchySorting` (illustrated in Fig. 8d) exploits heuristic `RandomMethod` as a starting point and proceeds by eliminating as many crossings as possible.

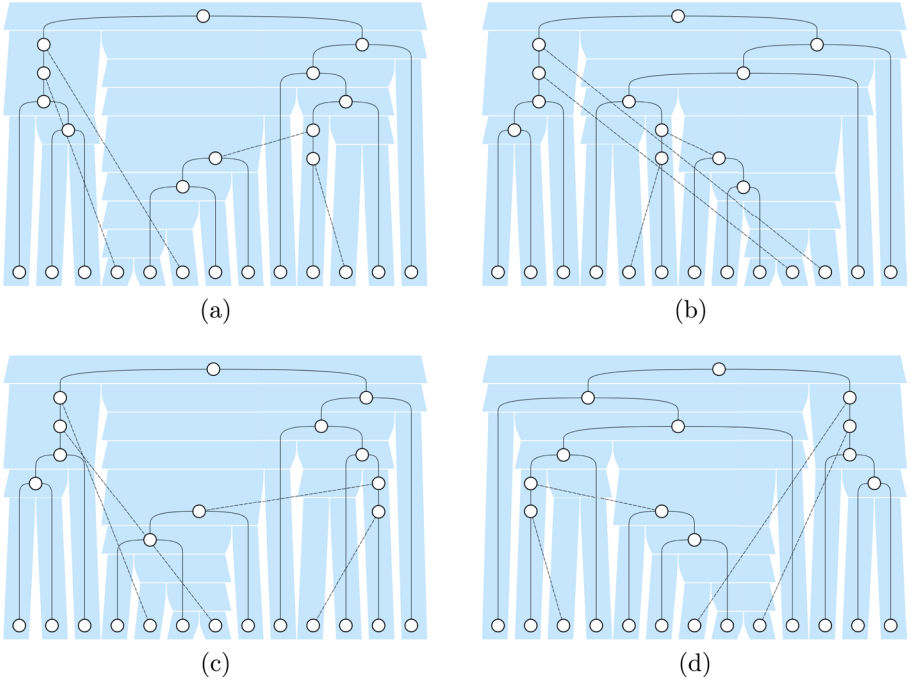


Fig. 8 Output of VIRI for a reconciliation of *Rodents & Pinworms* [22], when the used heuristic is: **a** `ShortenHostSwitches`; **b** `SearchMaximalPlanar`; **c** `RandomMethod`; **d** `HierarchySorting`

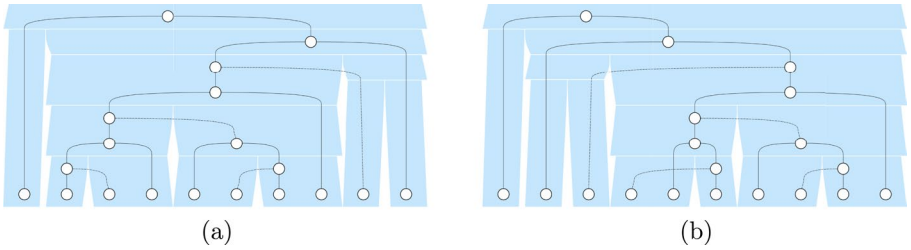


Fig. 9 Output of VIRI for a reconciliation of the planar instance *Anicetus* (*Hymenoptera*: *Encyrtidae*) parasitoids and their scale insect hosts (*Hemiptera Coccidae*) [26], when the used heuristic is: **a** either `ShortenHostSwitches` or `SearchMaximalPlanar` or `HierarchySorting`; **b**`RandomMethod`

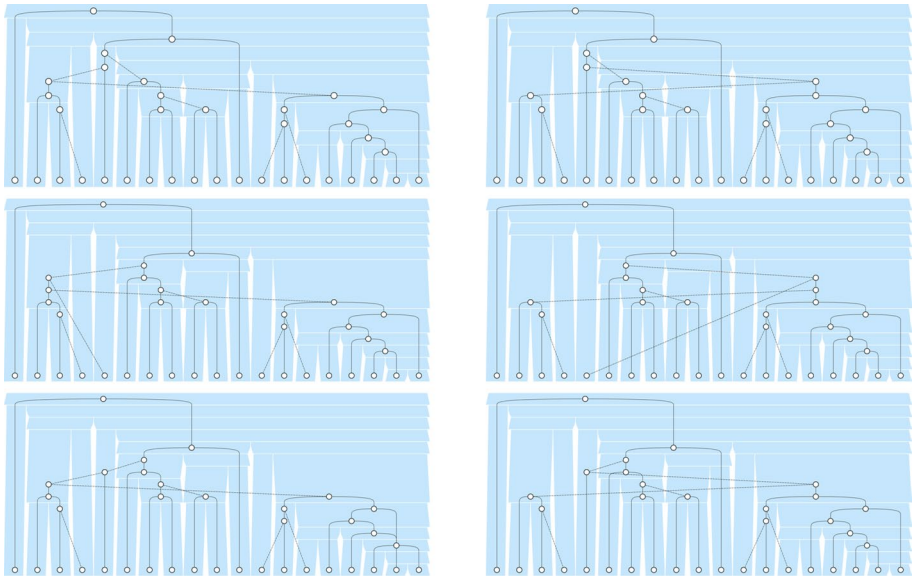


Fig. 10 The output of VIRI for the six reconciliations of *Cichlidogyrus & Tropheini* [27]

More specifically, it is inspired by Sugiyama's work on layered graphs [25], which aims to reduce the number of crossings by adjusting the order of nodes within each layer.

It is worth noting that before this heuristic is executed, `SearchMaximalPlanar` is called. If G is found to be planar, this heuristic is not run, and the planar drawing of G produced by `SearchMaximalPlanar` is output.

In Fig. 9a, the representation of a planar instance is depicted; it is the result of the heuristics `SearchMaximalPlanar`, `ShortenHostSwitches` and `HierarchySorting`; instead, a possible output of heuristic `RandomMethod` is shown in Fig. 9b and in general is not planar, due to the introduction of the randomness.

Finally, `ShowMultipleReconciliations` is designed to handle input containing multiple reconciliations by exploiting the `SearchMaximalPlanar` heuristic. It embeds the host tree only once, and thus maintains a consistent host tree drawing across all reconciliations. This approach is particularly useful for users comparing different reconciliations of the same instance, as it allows them to focus only on the changes in the symbiont tree's drawing, thus preserving their mental map (see Fig. 10).

Observe that, if the graph G defined above is planar, all the reconciliations to be visualized will be planar, in view of a result in [20].

We conclude this section by describing the architecture of VIRI depicted in Fig. 11. Depending on the requested URL, the web server calls the appropriate Python function through the web server gateway interface (WSGI) [28]. The Python logic is responsible for storing and retrieving information from the PostGreSQL database, where the information on the users and the available data files are recorded. The input `.nex` files describing cophylogeny instances and the output `.out` files describing the corresponding reconciliations are stored in the server file system, in suitable sub-directories (one for each user). The computation of the actual reconciliations' drawings is performed in two different ways, depending on the heuristics at hand. When the drawing heuristics do not involve a graph planarity test step (heuristics `ShortenHostSwitches`, `RandomMethod`, and `HierarchySorting`), the drawing is computed by the client Web browser *via* algorithms implemented in JavaScript. When, instead, the heuristics involve a graph planarity test (heuristics `SearchMaximalPlanar` and `ShowMultipleReconciliations`), the drawing is computed server-side. In this case, the Python logic exploits the GDToolkit C++ library [23, 24] by saving suitable input files in the file system, calling GDToolkit to compute the planar embedding of the instance, and then retrieving the output from the file system again. In both cases, the drawing is shown to the user using the JavaScript libraries Bootstrap [29, 30] and D3.js [31, 32].

Results and discussion

VIRI (visual inspector of reconciliation instances) is a new tree reconciliation visualizer. It takes as input a pre-computed reconciliation list, that needs to be visualised, in the .out format produced by the Capybara tool. Reconciliations generated by other tools can also be used, provided a simple conversion script is written to match the Capybara .out format. We also require the corresponding nexus file containing the host tree, symbiont tree and the mapping between symbiont and host tips, mainly for dataset storage and reference. Then it combines space-filling techniques for the host tree with node-link diagrams for the symbiont tree. This approach offers an intuitive and engaging way to explore co-evolutionary relationships between hosts and symbionts. By placing symbiont nodes inside the rectangles corresponding to the host they are mapped to, the reconciliation is clearly conveyed without overcrowding symbiont nodes in a confined space.

The proposed metaphor is designed to quickly convey all the information: the rectangles representing the host nodes are slanted to suggest which are the left and right children; all the reconciliation events can be easily detected:

- A loss corresponds to a parasite edge traversing a host rectangular node - see the edge connecting the two red nodes in Fig. 12a;
- In the presence of duplication, both the children go toward the same host rectangular child - see the green nodes in Fig. 12a;

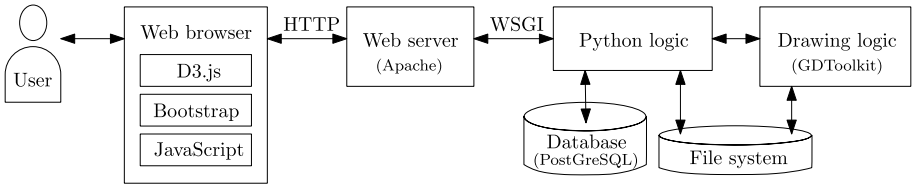


Fig. 11 VIRI architecture

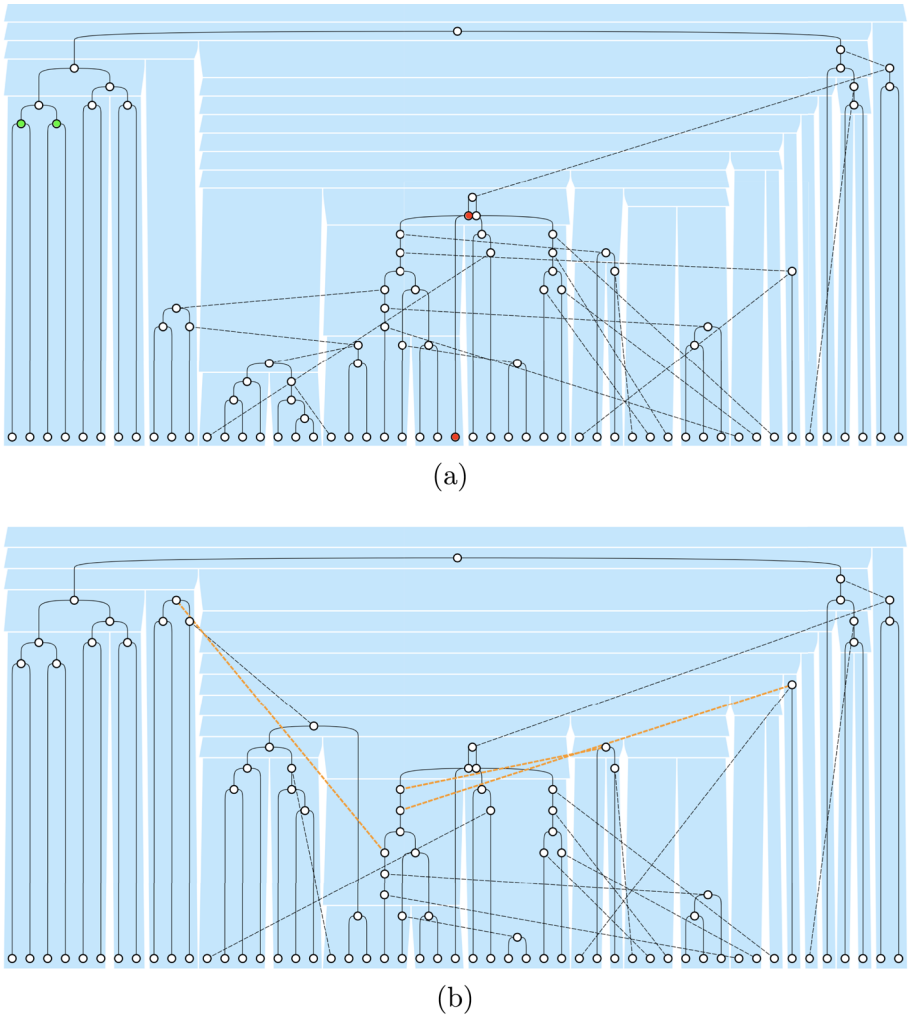


Fig. 12 Illustrative example based on VIRI output for two reconciliations of *Fishes & Dactylogyrus* [33]. In **a**, loss and duplication events are manually highlighted in red and green, respectively, for explanatory purposes only; in **b**, orange dashed edges manually added to indicate host-switches introducing time inconsistency

- Host-switches are represented as dashed lines, highlighted in orange if they are recognized to introduce a time inconsistency–see Fig. 12b.

Moreover, VIRI tries to minimize the crossing number, maximizing the clearness of the reading.

Finally, as demonstrated in Fig. 10, VIRI enables users to compare multiple reconciliations of the same host-symbiont tree instance. Unlike other tools, it preserves the user's mental map by keeping unaltered the representation of the host tree. The host tree representation is designed to minimize the total number of crossings across all reconciliations considered.

It is worth noting that the visualization algorithms are rather time-consuming, so they are fast when the host and symbiont trees are not too big (order of a few dozen nodes); nevertheless, VIRI produces easily-readable visualizations even for larger instances (see, for example, Fig. 13).

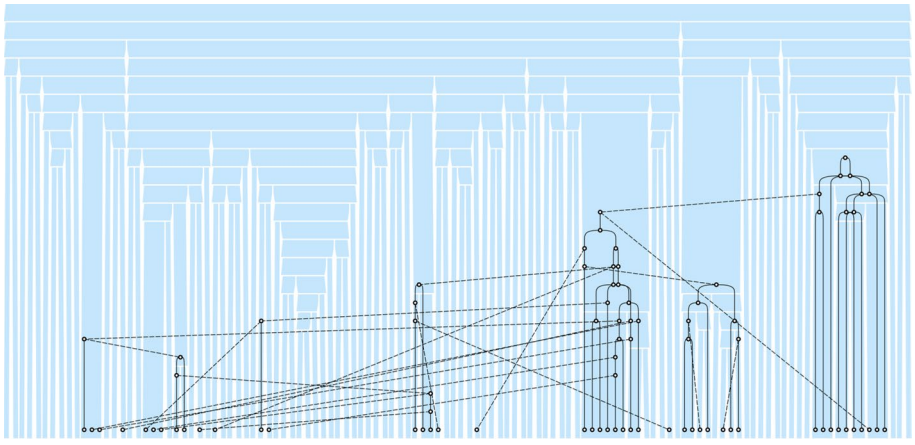


Fig. 13 The output of VIRI for a reconciliation of *COG* [34], whose host tree has 100 nodes and symbiont tree has 44 nodes

Conclusions

VIRI is a web application allowing users to visualize their own reconciliations, as output by any tool reconciling pairs of trees compatible with the .out format produced by the Capybara. It is an interactive tool and stimulates the sharing of reconciliations for a common and faster contribution to the field of cophylogenetics. It is designed with optimization in mind; indeed, it is possible to keep the number of crossings small, the host switches short, and the time to manually compare reconciliations fast by keeping the mental map in case of multiple reconciliations.

In the future, we plan to add several new features and improvements. In particular, we plan to give the user the possibility of editing the drawing within the constraints imposed by the layout. For example, it could be useful to allow repositioning of symbiont tree nodes within the rectangle representing their associated host node.

We also plan to support drawing customization by incorporating additional information, such as clearly displaying tip names (with options for abbreviation), and showing more informative content on hover, such as reconciliation-derived events on the symbiont tree. Further improvements include information such as geographical data visualized through color coding.

We will also improve the interface so that interaction is not needed to disambiguate the drawing when an edge-node overlap is present.

To address the long loading times observed for large datasets with a significantly high number of reconciliations, we are considering storing, along with the .nex and .out files, at least one precomputed visualization to avoid recomputing it from scratch each time. Based on user feedback, we will decide whether to store the initial optimal layout or a version that includes the user's modifications, and redesign VIRI to either load this saved layout or recompute the visualization as needed. Improving VIRI's efficiency under these conditions remains a key priority for future development.

Finally, while VIRI currently supports reconciliations in the Capybara .out format, reconciliations from other tools can be used as well, provided a simple conversion script is written to match this format. We plan to extend VIRI to alternative models of cophylogeny, such as those using node-to-edge mappings or other macro-evolutionary events, in future versions.

Availability and requirements

Project name: VIRI - Visual Inspector of Reconciliation Instances
Project home page: https://viri.di.uniroma1.it/
Operating system(s): Platform independent
Programming language: Javascript browser's interpreter
License: The VIRI service is provided "as is" and is freely available on the Web at https://viri.di.uniroma1.it/, without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose, and non-infringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the service or the use or other dealings in the service. The source code is available upon request under the terms of the GNU General Public License v3, with the exception of the components Apache2, PostgreSQL, and the GDToolkit Library that must be acquired independently
Any restrictions to use by non-academics: no restriction.

Acknowledgements
VIRI benefited from the work contribution of the following (Bachelor, Master, or Ph.D.) students: Valentino Di Donato, Carlo Iurato, Luca Lauro, Diego Mariottini, and Riccardo Paparozzi.

Author contributions
All authors have contributed to the paper's content, as outlined below: TC, MP, BS: Conceptualization; GD: Software; MP: Supervision; TC, MP: Methodology; TC, MP, BS: Validation; TC: Funding acquisition; TC, MP, BS: Writing.

Funding
This research was supported in part by MUR PRIN Project EXPAND, grant number 2022TS4Y3N, and in part by Sapienza University of Rome, grant number RM1241905ECC7E82.

Data availibility
This paper does not rely on external datasets. The web service associated with this work nevertheless includes only for illustrative purposes four datasets commonly used in host-symbiont cophylogeny studies. These datasets, together with their reconciliation outputs, are provided at https://viri.di.uniroma1.it/examples_table_visualization and can be downloaded in standard formats (.nex, .out). The corresponding reference papers for these datasets are also cited on the website.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 5 July 2024 / Accepted: 28 August 2025
Published online: 08 October 2025

References

1. Charleston MA. Recent results in cophylogeny mapping. Adv Parasitol. 2003;54:303–30. https://doi.org/10.1016/s0065-308x(03)54007-6.
2. Merkle D, Middendorf M. Reconstruction of the cophylogenetic history of related phylogenetic trees with divergence timing information. Theory Biosci. 2005;123:277–99. https://doi.org/10.1016/j.thbio.2005.01.003.
3. Donati B, Baudet C, Sinaimeri B, Crescenzi P, Sagot M. Eucalypt: efficient tree reconciliation enumerator. Algorithm Mol Biol. 2015;10(1):3. https://doi.org/10.1186/s13015-014-0031-3.
4. Santichaivekin S, Yang Q, Liu J, Mawhorter R, Jiang J, Wesley T, et al. eMPRess: a systematic cophylogeny reconciliation tool. Bioinformatics. 2020;37(16):2481–2. https://doi.org/10.1093/bioinformatics/btaa978.
5. Wang Y, Mary A, Sagot M-F, Sinaimeri B. Capybara: equivalence ClAss enumeration of coPhylogenY event-BAsed Reconcili-Ations. Bioinformatics. 2020;36(14):4197–9. https://doi.org/10.1093/bioinformatics/btaa498.
6. Menet H, Daubin V, Tannier E. Phylogenetic reconciliation. PLOS computational. Biology. 2022;18(11):1010621. https://doi.org/10.1371/journal.pcbi.1010621.

7. Wieseke N, Hartmann T, Bernt M, Middendorf M. Cophylogenetic reconciliation with ILP. IEEE/ACM Trans Comput Biol Bioinf. 2015;12(6):1227–35. https://doi.org/10.1109/tcbb.2015.2430336.
8. Conow C, Fielder D, Ovadia Y, Libeskind-Hadas R. Jane: a new tool for the cophylogeny reconstruction problem. Algorithm Mol Biol. 2010;5(1):16. https://doi.org/10.1186/1748-7188-5-16.
9. Merkle1 D, MM, Wieseke N. CoRe-PA. https://siks.informatik.uni-leipzig.de/software/core-pa/core-pa_tutorial.pdf.
10. Zhao P, Liu F, Li Y-M, Cai L. Inferring phylogeny and speciation of gymnosporangium species and their coevolution with host plants. Sci Rep. 2016;6(1):29339. https://doi.org/10.1038/srep29339.
11. Donati B, Baudet C, Sinaimeri B, Crescenzi P, Sagot M-F. EUCALYPT: efficient tree reconciliation enumerator. Algorithm Mol Biol. 2015;10(1):3. https://doi.org/10.1186/s13015-014-0031-3.
12. Comte N, Morel B, Hasić D, Guéguen L, Boussau B, Daubin V, et al. Treerecs: an integrated phylogenetic tool, from sequences to reconciliations. Bioinformatics. 2020;36(18):4822–4. https://doi.org/10.1093/bioinformatics/btaa615.
13. Kuitche E, Qi Y, Tahiri N, Parmer J, Ouangraoua A. DoubleRecViz: a web-based tool for visualizing transcript-gene-species tree reconciliation. Bioinformatics. 2020;37(13):1920–2. https://doi.org/10.1093/bioinformatics/btaa882.
14. Sennblad B, Schreil E, Sonnhammer A-CB, Lagergren J, Arvestad L. Primetv: a viewer for reconciled trees. BMC Bioinform. 2007;8(1):148. https://doi.org/10.1186/1471-2105-8-148.
15. Penel S, Menet H, Tricou T, Daubin V, Tannier E. Thirdkind: displaying phylogenetic encounters beyond 2-level reconciliation. Bioinformatics. 2022;38(8):2350–2. https://doi.org/10.1093/bioinformatics/btac062.
16. Chevenet F, Doyon J-P, Scornavacca C, Jacox E, Jousselin E, Berry V. SylvX: a viewer for phylogenetic tree reconciliations. Bioinformatics. 2015;32(4):608–10. https://doi.org/10.1093/bioinformatics/btv625.
17. Kruskal JB, Landwehr JM. Icicle plots: better displays for hierarchical clustering. Am Stat. 1983;37(2):162. https://doi.org/10.2307/2685881.
18. Kaya G, Ezekannagha C, Heider D, Hattab G. Context-aware phylogenetic trees for phylogeny-based taxonomy visualization. Front Genet. 2022;13:891240. https://doi.org/10.3389/fgene.2022.891240.
19. Lemieux S, Sargeant T, Laperrière D, Ismail H, Boucher G, Rozendaal M, et al. MiSTIC, an integrated platform for the analysis of heterogeneity in large tumour transcriptome datasets. Nucleic Acids Res. 2017;45(13):122–122. https://doi.org/10.1093/nar/gkx338.
20. Calamoneri T, Di Donato V, Mariottini D, Patrignani M. Visualizing co-phylogenetic reconciliations. Theoret Comput Sci. 2020;815:228–45. https://doi.org/10.1016/j.tcs.2019.12.024.
21. Tofigh A, Hallett M, Lagergren J. Simultaneous identification of duplications and lateral gene transfers. IEEE ACM Trans Comput Biol Bioinf. 2011;8(2):517–35. https://doi.org/10.1109/TCBB.2010.14.
22. Hugot J. New evidence for hystricognath rodent monophyly from the phylogeny of their pinworms. Tangl Trees Phylogeny Cospeciation Coevol. 2003; pp. 144–173.
23. GDToolkit - An object-oriented C++ library for handling and drawing graphs. http://gdt.inf.uniroma3.it/gdt4/index.php. Accessed 21 June 2025.
24. Di Battista G, Didimo W. GDToolkit. In: Tamassia R, editor. Handbook on graph drawing and visualization. Boca Raton: Chapman and Hall/CRC; 2013. p. 571–97.
25. Sugiyama K, Tagawa S, Toda M. Methods for visual understanding of hierarchical system structures. IEEE Trans Syst Man Cybern. 1981;11(2):109–25. https://doi.org/10.1109/TSMC.1981.4308636.
26. Deng J, Yu F, Li H-B, Gebiola M, Desdevises Y, Wu S-A, et al. Cophylogenetic relationships between anicetus parasitoids (hymenoptera: Encyrtidae) and their scale insect hosts (hemiptera: Coccidae). BMC Evol Biol. 2013;13:275. https://doi.org/10.1186/1471-2148-13-275.
27. Vanhove MPM, Pariselle A, Van Steenberge M, Raeymaekers JAM, Hablützel PI, Gillardin C, et al. Hidden biodiversity in an ancient lake: phylogenetic congruence between lake tanganyika tropheine cichlids and their monogenean flatworm parasites. Sci Rep. 2015;5(1):13669. https://doi.org/10.1038/srep13669.
28. Eby PJ. PEP 3333 - Python Web Server Gateway Interface v1.0.1. https://peps.python.org/pep-3333/. Accessed 16 June 2024 2010.
29. Build fast, responsive sites with Bootstrap. https://getbootstrap.com/. Accessed 16 June 2024.
30. Efron B, Tibshirani RJ. An introduction to the bootstrap. Monographs on statistics and applied probability, Chapman & Hall/CRC, Boca Raton. 1993; vol. 57.
31. Bostock M. D3.js - Data-driven documents. http://d3js.org/. Accessed 16 June 2024 2012.
32. Bostock M, Ogievetsky V, Heer J. D3: data-driven documents. IEEE Trans Vis Comp Graphics (Proc InfoVis). 2011;17(12):2301–9.
33. Balbuena JA, Míguez-Lozano R, Blasco-Costa I. Paco: a novel procrustes application to cophylogenetic analysis. PLoS ONE. 2013;8(4):61048. https://doi.org/10.1371/journal.pone.0061048.
34. David LA, Alm EJ. Rapid evolutionary innovation during an archaean genetic expansion. Nature. 2010;469(7328):93–6. https://doi.org/10.1038/nature09649.

Publisher's Note