Evolution of gene neighborhoods within reconciled phylogenies Supplementary Information

Sèverine Bérard, Coralie Gallien, Bastien Boussau,
Gergely J. Szöllősi, Vincent Daubin and Eric Tannier

Appendix 1: The chronology of duplications

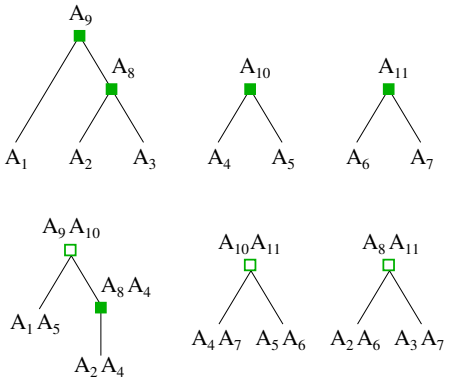

Figure S1: Consider the 3 genes trees at the top of this figure and the 6 following adjacencies : A_1A_5, A_5A_6, A_6A_2, A_2A_4, A_4A_7, A_7A_3, all genes belong to species A. The 3 adjacencies trees at the bottom of this figure are in an adjacency forest representing relationships between adjacencies. In this forest, A_8 and A_{11} duplicate together, as well as A_{11} and A_{10}, and A_{10} and A_9, so A_8 and A_9 should be contemporaneous, which contradicts the fact that $P(A_8) = A_9$.

Appendix 2: Correctness

The correctness of the algorithm depends on the proofs of three results. We first prove Lemma 1.

Proof. If AB and CD are in the same adjacency tree, then they share a common ancestor EF. By definition of an adjacency tree (points 6 to 8), E

is an ancestor of A and C, and F is an ancestor of B and D, which imposes the first property. The second also comes from the definition of the adjacency tree, imposing that if $A(N) = XY$ for a node N, then $S(X) = S(Y)$, therefore $S(E) = S(F)$. Suppose now A and B are in the same gene tree. C and D also are in this same gene tree by the first property. This means that E and F are in this same gene tree. The lowest common ancestor of E and F is also the one of A and B, and the one of C and D. As $S(E) = S(F)$, their lowest common ancestor is a gene duplication node. QED

Then we prove that the algorithm provides a result with the good properties, and to finish that the result is optimal.

Theorem S1 *The algorithm outputs an adjacency forest.*

Proof.

Claim. At least one of $c_1(AB)$ and $c_0(AB)$ is finite.

For **Cases 1.**, **2.**, **3.**, it is trivially the case.

For all others cases except **Case 6.(D12)**, when a call to some sum of $c_i(AB)$ and $c_j(CD)$ $(i, j \in \{0, 1\})$ is made, then the all four combinations are computed ($c_0(AB)+c_0(CD)$, $c_0(AB)+c_1(CD)$, $c_1(AB)+c_0(CD)$ and $c_1(AB)+c_1(CD)$). So by recurrence one of the four is finite.

For the **Case 6.(D12)**, each line is the sum of 4 costs, as all the 16 combinations are computed, by recurrence, one of it is finite.

Claim. The algorithm gives a forest

Every call to c_1 creates a vertex. In **Cases 4.** to **6.**, every such vertex has one or two children, and if AB and CD are the children of a node, then A, B, C and D are not descendants of each other. In **Cases 1.** to **3.**, created vertices are leaves as well as nodes labeled "Break". So the result is actually a forest where each internal node has one or two children as required by the definition of an adjacency tree.

Claim Every node XY verifies $S(XY) = S(X) = S(Y)$.

This is trivial because all examined couples have this property.

Claim Each component of the forest is an adjacency tree.

We have to check every property of the adjacency tree:

1) It is evident by construction that if N is a leaf, then $E(N) \in \{Extant, GLos, ALos, Break\}$ and if N is an internal node, then $E(N) \in \{Spec, GDup, ADup\}$.

2) By construction also, if $E(N) \neq Break$, $A(N) = XY$ where X and Y are gene tree nodes such that $S(N) = S(X) = S(Y)$.

3) **Case 1.** ensures that if $E(N) = Extant$, then $G(X)G(Y)$ is an adjacency.

4) **Case 2.** ensures that if $E(N) = GLos$, then $E(X) = GLos$ or $E(Y) = GLos$ (and not both).

5) **Case 3.** ensures that If $E(N) = ALos$, then $E(X) = E(Y) = GLos$.

6) **Case 5.** ensures that If $E(N) = Spec$, then $E(X) = E(Y) = E(N)$, and that in addition, N has two children $N1$ and $N2$ and $A(N1)$ and $A(N2)$ are couples of children of X and Y.

7) **Case 6.(D12)** ensures that if $E(N) = ADup$, then $E(X) = E(Y) = GDup$, and that in addition, N has two children $N1$ and $N2$ and $A(N1)$ and $A(N2)$ are couples of children of X and Y.

8) **Cases 4.** and **6.(D1&D2)** ensure that if $E(N) = GDup$, then $E(X) = GDup$ or $E(Y) = GDup$ (suppose it is Y), and that in addition, N has only one child $N1$ and $A(N1)$ is a couple of genes composed of X and one child of Y.

Claim. Any adjacency XY between two leaves of two gene subtrees rooted at A and B is in the forest constructed by $c_1(AB)$ and $c_0(AB)$ when these are finite numbers.

By induction on $n+m$, the sum of the depths (maximum distance from the root to a leaf) of the two gene trees. If $n+m = 0$ then $XY = AB$ and the result is implied by **Case 1.** Suppose it is true for any number below $n + m$, and T_A and T_B are two subtrees of depth n and m. The algorithm systematically calls c_1 or c_0 on direct descendants of A or/and B. If X and Y are descendants of A and B, they are also descendants of one of the direct descendants. By induction the forest constructed from these descendants contains XY.

Claim. Adjacencies XY labeling the nodes of the forest are all distinct.

This is implied by the division in classes. Any adjacency (extant or ancestral) can be assigned to an equivalence class. For extant ones, it is contained in the definition, and for an ancestral adjacency XY, it is assigned to the class where there is an extant adjacency AB, and A is a descendant of X, and B is a descendant of Y. It is unambiguous from the definition of the classes: if there are two such extant adjacencies AB, they belong to the same class. So each adjacency XY belongs to exactly one class and as the backtracking procedure doesn't examine twice the same pair of gene tree nodes; it doesn't construct twice a node labeled with the same adjacency.

Eventually all properties of the adjacency forest are fulfilled. QED

Now we give the proof of Theorem 1 from the main text.

Proof. We prove the theorem by induction on $n + m$, the sum of the depths (maximum distance from the root to a leaf) of the two gene trees.

If $n + m = 0$, the cost of an optimal adjacency forest is trivially 0.

Suppose now that it is true for any number below $n + m > 0$. Now we have two trees T_1 and T_2 of depth n and m, rooted at vertices $R1$ and $R2$.

It is not possible that $E(R1R2) = Extant$ or $E(R1R2) = ALos$ because $n + m > 0$. If $E(R1R2) = GLos$, then the cost of an optimal adjacency forest is trivially 0.

Now otherwise let $c1opt$ be the minimum cost of an adjacency forest $Fopt$ between $T1$ and $T2$, containing $R1R2$.

If $E(R1R2) = Spec$, then from the definition of an adjacency tree, $R1$ has two descendants $R1a$ and $R1b$, and $R2$ has two descendants $R2a$ and $R2b$, such that $S(R1a) = S(R2a)$ and $S(R1b) = S(R2b)$.

Let F' be constructed from $Fopt$ by removing $R1R2$ and its children labeled by breakages.

Then F' is the the union of an optimal adjacency forest on $T1(R1a)$ and $T2(R2a)$, containing $R1aR2a$ or not according to if $Fopt$ contains it or not, and an optimal adjacency forest on $T1(R1b)$ and $T2(R2b)$, containing $R1bR2b$ or not according to if $Fopt$ contains it or not. Indeed, if not, take these optimal solutions, add $R1R2$ and get a solution with a lower cost than $Fopt$.

By induction, the algorithm is able to construct F', and **Case 5.** constructs $Fopt$ from F'.

Now if $E(R1R2) = GDup$, then from the definition of an adjacency tree, $R1R2$ has one child, while (say) $R1$ is a duplication and has two children $R1a$ and $R1b$.

Let F' be constructed from $Fopt$ by removing $R1R2$ and its child if labeled by a breakage.

Then F' is the the union of an optimal adjacency forest on $T1(R1a)$ and $T2(R2)$, containing $R1aR2$ or not according to if $Fopt$ contains it or not, and an optimal adjacency forest on $T1(R1b)$ and $T2(R2)$, containing $R1bR2$ or not according to if $Fopt$ contains it or not. Indeed, if not, take these optimal solutions, add $R1R2$ and get a solution with a lower cost than $Fopt$.

By induction, the algorithm is able to construct F', and **Case 4.** or **6.(D1&D2)** constructs $Fopt$ from F'.

If $E(R1R2) = ADup$, then $R1$ and $R2$ are both duplication nodes, and have descendants $R1a$, $R1b$, $R2a$, $R2b$ labeled by the same species.

Let F' be constructed from $Fopt$ by removing $R1R2$ and its children labeled by breakages.

Then F' is the the union of four optimal adjacency forests: (1) on $T1(R1a)$ and $T2(R2a)$, containing $R1aR2a$ or not according to if $Fopt$ contains it or not, (2) on $T1(R1a)$ and $T2(R2b)$, containing $R1bR2b$ or not according to if $Fopt$ contains it or not, (3) on $T1(R1b)$ and $T2(R2a)$, containing $R1aR2a$ or not according to if $Fopt$ contains it or not, (4) on $T1(R1b)$ and $T2(R2b)$, containing $R1bR2b$ or not according to if $Fopt$ contains it or not. Indeed, if not, take these optimal solutions, add $R1R2$ and get a solution with a lower cost than $Fopt$.

By induction, the algorithm is able to construct F', and **Case 6.(D12)** constructs $Fopt$ from F'.

This proves that the algorithm computes an optimal solution in the c_1 case.

Now let $c0opt$ be the minimum cost of an adjacency forest $Fopt$ between $T1$ and $T2$, not containing $R1R2$.

In this case $Fopt$ is the union of two (if $E(R1R2) = Spec$ or $E(R1R2) = GDup$) or four (if $E(R1R2) = ADup$) optimal adjacency forests on the children of $R1$ and/or $R2$. By induction an equivalent is found by the algorithm. QED

Appendix 3: The alternative method to infer ancestral adjacencies

We implemented an alternative method in the spirit of [3], [1] or [2], who all perform pairwise genome comparisons and merge ancestral adjacencies from these pairs. The following formalization is used in [2] to assess the quality of gene trees.

We search for all quadruples G,H,I,J of extant genes, such that:

- G and H are adjacent in the genome of an extant species S_1;
- I and J are adjacent in the genome of an extant species S_2 different from S_1;
- the lowest common ancestor A_1 of G and I is a speciation node and belongs to an ancestral species S_A;
- the lowest common ancestor A_2 of H and J is distinct from A_1 and belongs to S_A.

For each such quadruple G,H,I,J, we draw adjacencies between all pairs of ancestral genes tree nodes X and Y such that:

- $X = A_1$ and $Y = A_2$, or
- $S(X) = S(Y) = S_B$ is a descendant of S_A, and X is an ancestor of G, and Y is an ancestor of H, or
- $S(X) = S(Y) = S_B$ is a descendant of S_A, and X is an ancestor of I, and Y is an ancestor of J.

Appendix 4: Compared phylogenies of mammals and angiosperms

See Figures S2 and S3. All phylogenies are obtained with the datasets three and four, with the same procedure as the one used to draw Figure 4. Only the branch length computations have different meanings.

References

[1] D. Bertrand, Y. Gagnon, M. Blanchette, and N. El-Mabrouk. Reconstruction of ancestral genome subject to whole genome duplication, speciation, rearrangement and loss. In *Algorithms in Bioinformatics, proceedings of WABI'10*, Lecture Notes in Bioinformatics. Springer, 2010.

[2] B. Boussau, G. Szollosi, L. Duret, M. Gouy, E. Tannier, and V. Daubin. Genome-scale coestimation of species and gene trees. *Submitted*, 2012.

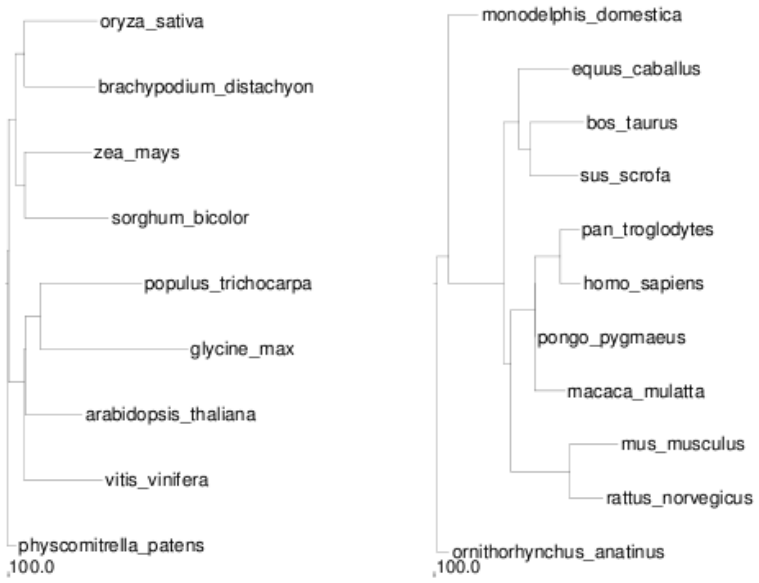

Figure S2: Angiosperm and Mammalian phylogenies, where branch lengths are proportional to the the number of adjacency gains.

[3] M. Muffato, A. Louis, C.-E. Poisnel, and H. R. Crollius. Genomicus: a database and a browser to study gene synteny in modern and ancestral genomes. *Bioinformatics*, 26(8):1119–1121, Apr 2010.

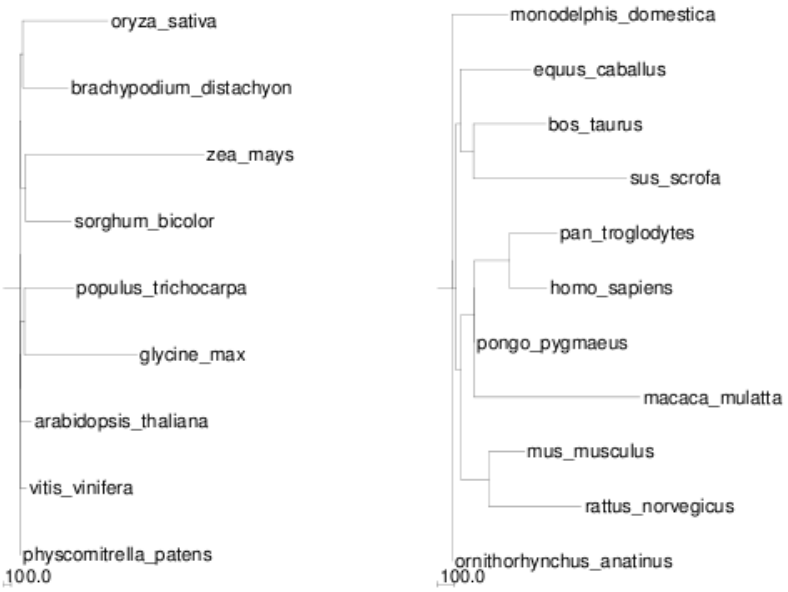

Figure S3: Angiosperm and Mammalian phylogenies, where branch lengths are proportional to the the number of adjacency breakages.