APPENDIX

Proof of Theorem 1: Let H be a minimum refinement of G. Then H is a binary arrangement of the n subtrees H_i for $1 \leq i \leq n$, where each H_i is a refinement of G_i. Suppose that for a given i, H_i is not a minimum refinement of G_i. Then replacing H_i by $H_{min}(G_i, S)$ lowers the number of NAD nodes in the subtree rooted at x_i, but has no effect on the type of nodes outside this subtree. It follows that a minimum refinement of G is a minimum refinement of G'. □

Proof of Lemma 1: Sufficiency is clearly true. As for necessity, let J be a join sequence with d NADs, and $J_{i-1} = \{G_1, G_3\}$ be the first join of type AD or NAD in J followed by a join $J_i = \{G_2, G_4\}$ of type S. We show that we can swap the join types of J_{i-1} and J_i. In other words, we create a new sequence J' where $J'_k = J_k$ for $k < i-1$, J'_{i-1} is of type S, J'_i of the same type as J_{i-1}, and all subsequent joins types are the same as in J. We can then apply this swapping procedure until all S joins are in the beginning of J'.

Let $G_{1,3}$ denotes the subtree created after applying J_{i-1}. If neither G_2 nor G_4 are equal to $G_{1,3}$, then we can safely swap J_{i-1} and J_i since they create two independent subtrees, which does not affect subsequent joins. So suppose w.l.o.g. that $G_4 = G_{1,3}$, and therefore $J_i = \{G_{1,3}, G_2\}$. Since J_i is of type S, then G_1 and G_3 both shared an S edge with G_2 in $\mathcal{F}(J, i-2)$. Let $J'_{i-1} = \{G_1, G_2\}$ be the join of type S, which creates a subtree denoted $G_{1,2}$, and let $J'_i = \{G_{1,2}, G_3\}$. If J_{i-1} is of type AD, then by applying Ruleset 1.1 (replacing T by G_3), it follows that $J'_i = \{G_{1,2}, G_3\}$ is of type AD. Conversaly, if J_{i-1} is of type NAD, then by applying Ruleset 1.2, it follows that J'_i if of type NAD. In other words, J'_i and J_{i-1} are of the same type. Since the subtrees created by applying J_i and J'_i share the same leafset, and that join types are defined by the leaves, all subsequent joins in J can be applied in J'. □

Proof of Theorem 2: We first prove a simple claim.

Claim i. Consider $H \in \mathcal{H}(G)$ with exactly d NADs. Then there exists a set W of vertex-disjoint cliques in R_S such that $R_{AD} \cup W$ has $d+1$ connected components.

Recall that d is the minimum number of NADs attainable. Let J be a join sequence realizing H. By Lemma 1, we can assume that all speciations precede all duplications. Let k be the number of maximum speciation subtrees of H. As stated before (statement just preceding the theorem), the set of leaves of each speciation subtree of H forms a clique in R_S. Moreover, as the k maximum speciation subtrees of H are disjoint (do not share a common node), the corresponding set W of cliques in R_S are vertex-disjoint. Let R^J be the graph obtained after applying all the speciations in J. If R^J has more than $d+1$ AD-components, then J cannot lead to a solution with d NADs. Indeed, we have exhausted all speciations used by J, which implies only NAD edges are used to join AD-components together - requiring more than d of them if there are more than $d+1$ AD-components. On the other hand, if R^J has less than $d+1$ AD-components, then there exists a solution with less than d NADs, contradicting the fact that d is the minimum number of NADs of a solution to the MinNADref Problem . It follows that R^J has exacly $d+1$ AD-components, which completes the proof of the claim.

"⇐" Let $d+1$ be the minimum number of connected components formed by the edges of R_{AD} augmented with the edges of a set W of vertex-disjoint cliques of R_S. Then all nodes of each connected component can be joined under a single subtree by applying joins of type AD and S. These $d+1$ subtrees can then be joined with exactly d NADs, yielding a refinement H with exactly d NADs. Then H is a solution to the MinNADref Problem as otherwise there is a refinement H^* with $d^* < d$ NADs, leading (Claim i) to a W^* such that $R_{AD} \cup W$ has $d* + 1 < d+1$ connected components, which contradicts the fact that $d+1$ is the minimum number of connected components formed by the edges of R_{AD} augmented with the edges of a set of vertex-disjoint cliques in R_S.

"⇒" Let H be a solution to the MinNADref Problem with d NADs. Then, by Claim i, there is a set W of vertex-disjoint cliques in R_S such that $R_{AD} \cup W$ has $d+1$ connected components. Now suppose that the minimum number of connected components induced by a set of vertex-disjoint cliques is $d^* + 1 < d+1$. By the sufficient proof above, it follows that d^* is the minimum number of NAD nodes of a resolution, contradicting the minimality of d.

□

Proof of Theorem 3: In this proof, for two vertices G_i, G_j of R, we denote $s_{i,j} = lca_S(s(G_i), s(G_j))$.

"⇒"Suppose R_S is not $\{P_4, 2K_2\}$-free. Let G_1, G_2, G_3, G_4 be four vertices, with $\{G_1, G_2\}$ and $\{G_3, G_4\}$ being two edges in R_S, that form an induced P_4 or $2K_2$. This implies that at least one of the two edges $\{G_1, G_3\}$ and $\{G_2, G_4\}$ should be absent from R_S.

Assume w.l.o.g. that $\{G_1, G_3\}$ is the missing edge. The edge between G_1, G_2 means that $s(G_1)$ and $s(G_2)$ are unrelated in $\mathcal{S}$. Suppose w.l.o.g. that $s(G_1)$ is in the left subtree of $s_{1,2}$, and $s(G_2)$ in the right subtree. The missing edge between G_1 and G_3 then implies that $s(G_1)$ and $s(G_3)$ are related, in other words that $s(G_3)$ is on the left subtree of $s_{1,2}$ or is an ancestor of $s_{1,2}$.

Suppose that $\{G_2, G_3\}$ is not an edge in R_S. Then, by a similar reasoning as before, it follows that $s(G_3)$ is either on the right subtree of $s_{1,2}$ or is an ancestor of $s_{1,2}$. It follows from the two arguments that $s(G_3)$ is an ancestor of $s_{1,2}$. Now, the edge between G_3, G_4 means that $s(G_3)$ and $s(G_4)$ are unrelated in $\mathcal{S}$, and thus $s(G_4)$ is unrelated to $s(G_1)$ and $s(G_2)$ as well. But in this case $\{G_1, G_4\}$ and $\{G_2, G_4\}$ should be edges in R_S, and thus G_1, G_2, G_3, G_4 can neither form a $2K_2$ nor a P_4 struture.

Now suppose that $\{G_2, G_3\}$ is an edge in R_S. Then G_1, G_2, G_3, G_4 cannot form a $2K_2$ structure, and for the four edges to form a P_4 structure, $\{G_2, G_4\}$ should not be an edge in R_S. By taking the same proof as above (switching G_3 and G_4), we have that $s(G_4)$ is an ancestor of $s_{1,2}$. Now, the edge $\{G_3, G_4\}$ means that $s(G_3)$ and $s(G_4)$ are unrelated in $\mathcal{S}$, and thus $s(G_3)$ is unrelated to $s(G_1)$ and $s(G_2)$ as well. But in this case $\{G_1, G_3\}$ should be an edge in R_S, and thus G_1, G_2, G_3, G_4 can neither form a $2K_2$ nor a P_4 structure.

In both cases, the evolutionary constraints on S edges in a valid graph lead to a contradiction with the assumption that R_S contains a P_4 or a $2K_2$. So if R_S contains a P_4 or a $2K_2$, R is not valid.

"⇐"The other direction of the proof uses the notion of cotrees, related to P_4-free graphs. A *cotree* T is a rooted tree in which the internal nodes are labelled 0 or 1 and have at least two children. We say T is an *alternating cotree* if the labels of any root-leaf path alternate between 0 and 1. A cotree T *represents* a given graph H if $l(T) = V(H)$, and $xy \in E(H)$ if and only if $lca_T(x, y)$ is

labelled by 1. It is well-known that for any P_4-free graph H, there is a unique alternating cotree that represents H. Let T denote the unique alternating cotree representing R_S. Note that $l(T) = V(R)$. The fact that R_S is also $2K_2$-free implies the following: any internal node x of T labelled 0 has at most one non-leaf child. If not, then x has two children x_1, x_2 labelled 1, which implies we can find $a, b \in l(x_1)$, $c, d \in l(x_2)$ such that $lca_T(a, b) = x_1$ and $lca_T(c, d) = x_2$. Since the lca of each pair $(a, c), (a, d), (b, c)$ and (b, d) is x, labelled 0, then a, b, c, d induces a $2K_2$ in which the edges are ab and cd.

Now, given R and T, we can construct a forest $\mathcal{F}$ and a species tree $\mathcal{S}$ that make R valid. Note that since T is constructed from R_S, for two leaves x, y of T, $lca_T(x, y)$ is labelled 1 if $jt(x, y) = S$, and labelled 0 if $jt(x, y) \in \{AD, NAD\}$. The reader may refer to Figure 1 for an example of the whole construction for a given R. The species tree is found by a transformation of T. Note that T is not necessarily binary, but the reader can verify that any binary refinement of the constructed species tree will result in a valid instance. Let $x \in l(T)$. We transform x into a bigger tree $\beta(x) = (\beta_{AD}(x), (x^*, \beta_{NAD}(x)))$, where x^* is a single leaf and $\beta_{AD}(x)$ and $\beta_{NAD}(x)$ are two copies of T. For some $y \in l(T)$, denote by $\beta_{AD}(x, y)$ (resp. $\beta_{NAD}(x, y)$) the unique leaf of $\beta_{AD}(x)$ (resp. $\beta_{NAD}(x)$) that corresponds to y in the copy.

The species tree $\mathcal{S}$ is obtained by replacing each leaf $x \in l(T)$ by $\beta(x)$. The point of $\beta(x)$ is to reserve the $\beta_{AD}(x)$ subtree for the vertices of R that x shares an AD relationship with, and the $\beta_{NAD}(x)$ subtree for the NAD relationship. Hence in $\mathcal{F}$, both trees corresponding to x and y will have a gene mapped to $\beta_{AD}(x, y)$ or to $\beta_{AD}(y, x)$ when $jt(x, y) = AD$. If $jt(x, y) = NAD$, then either y but not x will have a gene mapped to $\beta_{NAD}(x, y)$, or x but not y will have a gene mapped to $\beta_{NAD}(y, x)$

Denote by β_x the root of $\beta(x)$. Now on to the construction of $\mathcal{F}$. Let $x \in l(T)$. Let $\gamma(x)$ be a copy of $\beta(x)$ from which we remove the $\beta_{NAD}(x)$ subtree (hence $\gamma(x)$ is a copy of $(\beta_{AD}(x), x^*)$). The species that each gene of $\gamma(x)$ is mapped to is its corresponding leaf in $\beta(x)$. It follows from this that $s(\gamma(x)) = \beta_x$.

We finally construct $\mathcal{F}$ by adding a subtree for each $x \in l(T)$ as such :

- If the parent of x in T is labelled 1, add $\gamma(x)$ to $\mathcal{F}$.
- If the parent of x in T is labelled 0, start from $\gamma(x)$ and for each leaf y of T such that $lca_T(x, y)$ is the parent of x,
  - if $jt(x, y) = AD$, let $\gamma(x) \leftarrow (y', \gamma(x))$, where y' is a new gene such that $s(y') = \beta_{AD}(y, x)$.
  - if $jt(x, y) = NAD$, let $\gamma(x) \leftarrow (y', \gamma(x))$, where y' is a new gene such that $s(y') = \beta_{NAD}(y, x)$.

  then add the resulting $\gamma(x)$ to $\mathcal{F}$.

Note that from this, for $x \in l(T)$, if the parent of x is labelled 1 then $s(\gamma(x)) = \beta_x$, and if the parent of x is labelled 0, then $s(\gamma(x))$ is the parent of β_x, which is labelled 0. To see this, denote by $p(x)$ the parent of x in T, labelled 0. Observe that $p(x)$ remains unchanged in $\mathcal{S}$. Now, for each $y \in l(t)$ such that $lca_T(x, y) = p(x)$, $\gamma(x)$ has genes mapped to species of $\beta(y)$. Thus $\gamma(x)$ has only genes mapped to species that are descendants of $p(x)$ in $\mathcal{S}$, and thus $s(\gamma(x)) = p(x)$ in $\mathcal{S}$.

In both cases, $s(\gamma(x))$ is a descendant of its lowest ancestor labelled 1, if any. From this, we get that if x, y share an S edge in R, then they are left and right descendants of $lca_T(x, y)$ labelled 1, implying that $s(\gamma(x))$ and $s(\gamma(y))$ are left and right descendants of this same node labelled 1 in $\mathcal{S}$. They are therefore related by speciation as prescribed. Now, suppose that x, y are related by a NAD edge. If $lca_T(x, y)$ is $p(x)$, then $\gamma(x)$ contains y' mapped to $\beta_{NAD}(y, x)$. This forces $\gamma(x)$ and $\gamma(y)$ to be related by duplication, which is of type NAD since by construction $\gamma(x)$ and $\gamma(y)$ contain no gene mapped to the same species. The same argument holds when $lca_T(x, y)$ is $p(y)$. So suppose that $lca_T(x, y)$ is not $p(x)$ nor $p(y)$. Then $lca_T(x, y) = lca_T(p(x), p(y))$ and is labelled 0. But this implies that $lca_T(p(x), p(y))$ has at least two non-leaf children, one containing $p(x)$ and the other containing $p(y)$, contradicting the $2K_2$-free assumption as stated above. We observe that the same applies to vertices x, y of R related by an AD edge, except that they must share a gene mapped to $\beta_{AD}(y, x)$ or $\beta_{AD}(x, y)$, making them related by apparent duplication. We finally note that no other tree of $\mathcal{F}$ has genes mapped to $\beta_{AD}(y, x)$ or $\beta_{AD}(x, y)$, thereby removing the possibility of an unwanted apparent duplication.

□

Before being able to prove Theorem 4, we need the following general property on P_4-free graphs.

LEMMA 1. *Let $\{x, y\}$ be an edge of a P_4-free graph $\mathcal{G}$, and let W_x and W_y be two vertex-disjoint cliques of $\mathcal{G}$ respectively containing x and y. Then we can partition the vertices $V(W_x) \cup V(W_y)$ into at most two other cliques, with one containing $\{x, y\}$.*

PROOF. If the set $V(W_x) \cup V(W_y)$ induces a single clique, then we are done. Otherwise, let $Y(x) \subseteq V(W_y)$ denote the set of vertices in W_y that share an edge with x (including y), and let $X(Y) \subseteq V(W_x)$ be the vertices of W_x that share an edge with every vertex of $Y(x)$. The set $V_1 = \{x\} \cup Y(x) \cup X(Y)$ induces a clique containing $\{x, y\}$. Now, let a and b be two vertices sharing an edge of W_x and W_y respectively with x and y, such that $a, b \notin V_1$. If a, b are both in W_x, or both in W_y, then they obviously share an edge. Otherwise, suppose w.l.o.g. that a is in W_x and b in W_y. Because $a \notin X(Y)$, there is some $b_i \in Y(x)$ such that $\{a, b_i\} \notin E(\mathcal{G})$. And because $b \notin Y(x)$, $\{x, b\} \notin E(\mathcal{G})$. But $\{a, x, b_i, b\}$ induces a P_4, unless $ab \in E(G)$. Therefore, every pair of vertices in W_x or W_y but not in V_1 share an edge, forming our second clique.

If $Y(x)$ is empty, we can apply the same argument by symmetry using $X(y)$ and $Y(X)$ if $X(y)$ is not empty. If both $Y(x)$ and $X(y)$ are empty, then let $V_1 = \{x, y\}$ induce the first clique. Let a, b be vertices sharing edges with x and y respectively. Now, a, x, y, b induce a P_4 unless $\{a, b\} \in E(G)$, and thus second clique is formed by the vertices sharing an edge with x, y. □

Let c_{AD} be the number of AD-components of R before applying any join. Suppose we have a join sequence with s useful speciations, all applied before any AD or NAD join. It follows that applying a useful speciation connects two AD-components together, and applying s of them results in a graph with $AD_{AD} - s$ AD-components, from which we can obtain a tree with $d = AD_{AD} - s - 1$ NADs. It is then clear that there exists a solution with d NADs iff there exists a join sequence with $s = AD_{AD} - d - 1$ useful speciations. Hence we can minimize the number of NADs by maximizing the number of useful speciations we can make. Our

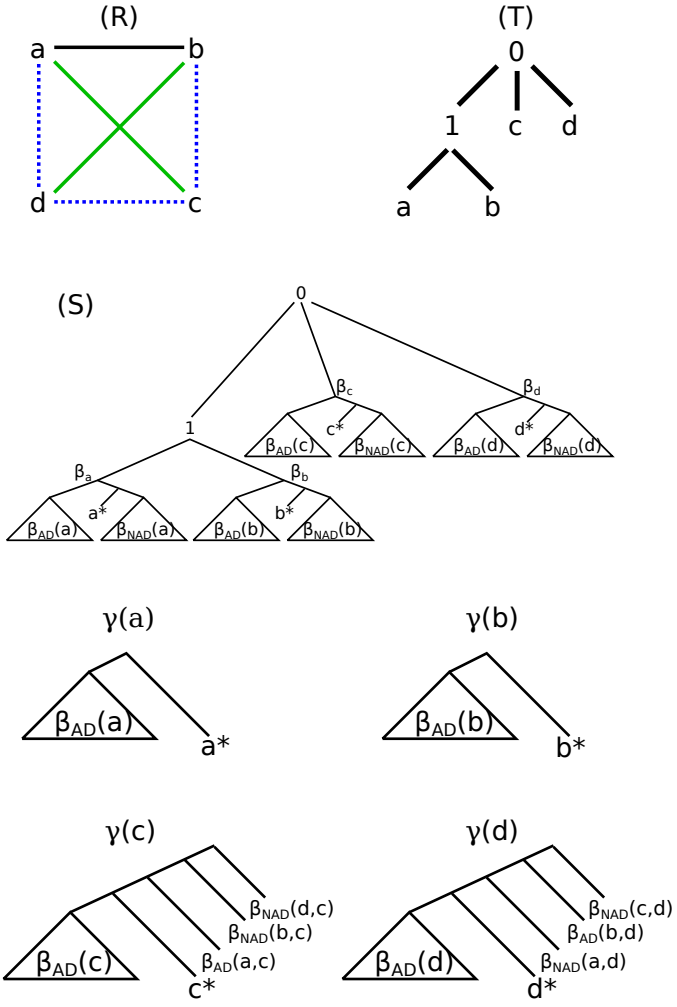

Fig. 1. A construction of $\mathcal{F}$ and $\mathcal{S}$ given R. The solid black edge of R is an S-edge, the green edges are AD-edges and the blue dotted edges are NAD-edges. T is the cotree corresponding to R_S, where $V(R) = l(T)$. The species tree $\mathcal{S}$ is build by replacing each leaf x of T by $\beta(x)$. For instance here, $\beta_{AD}(a)$ contains the leaves $\{\beta_{AD}(a,a), \beta_{AD}(a,b), \beta_{AD}(a,c), \beta_{AD}(a,d)\}$. The gene tree forest $\mathcal{F}$ consists of $\{\gamma(a), \gamma(b), \gamma(c), \gamma(d)\}$, in which we labelled the genes to their corresponding species, built from the construction given in the proof of Theorem 3.

heuristic consists in constructing a join sequence by always picking the lowest available speciation, which is shown to find at least half the number of useful speciations as the optimal solution. We first need the following property.

LEMMA 2. *Let $\{x, y\}$ be an S edge of R corresponding to a lowest available speciation, and let d be number of NADs of a solution to the* MinNADref *problem. Then there exists a solution which makes the $\{x, y\}$ speciation that has at most $d + 1$ NADs.*

PROOF. Let W be a set of vertex-disjoint cliques of R_S, and let R_W be the R graph restricted to the set of edges $W \cup R_{AD}$ (W must exist by Theorem 3). R_W has $d + 1$ connected components. Let W_x (resp. W_y) be the clique of W that contains x (resp. y). If $W_x = W_y$, then we are done. Otherwise, by Lemma 1, we can partition the vertices of W_x and W_y into two other cliques, namely W_1 containing the xy edge and the other clique W_2. Let $W' = W \setminus \{W_x, W_y\} \cup \{W_1, W_2\}$. Now, W' is another set of vertex-disjoint cliques. Denote by $R_{W'}$ the graph R restricted to $W' \cup R_{AD}$. Denote by Z_x, Z_y the vertices in $V(R) \setminus \{W_x, W_y\}$ in the same R_W component as x and y respectively. Similarly, let Z_1, Z_1 be the vertices in $V(R) \setminus \{W_1, W_2\}$ in the same $R_{W'}$ component as a vertex of W_1 and a vertex of W_2 respectively. We have that $Z_x \cup Z_y = Z_1 \cup Z_2$. If x, y were in two distinct components $W_x \cup Z_x$ and $W_y \cup Z_y$ in R_W, then $R_{W'}$ also has $d+1$ components, as these two components got replaced by $W_1 \cup Z_1$ and $W_2 \cup Z_2$. If x, y were in the same component, at worst $R_{W'}$ has $d+2$ components, having the x, y component replaced by $W_1 \cup Z_1$ and $W_2 \cup Z_2$. □

We are now ready to prove Theorem 4:

Proof of Theorem 4: Let $d = AD_{AD} - s - 1$ be the minimum number of NADs in an optimal solution, and let xy be the lowest useful speciation available in R. Note that $s = AD_{AD} - d - 1$. By Lemma 2, there exists a solution with $d + 1$ NADs that contains the xy speciation. Let R' be the graph obtained after applying the $\{x, y\}$ join, thus contracting x and y and applying Ruleset 1. Since xy is the lowest speciation, any common neighbor of x and y in R_S is a neighbor of the xy vertex in R'_S. Therefore, R'_S has $AD_{AD} - 1$ AD-components and admits an optimal solution with at most d NADs. Hence, the number of useful speciations we can make given R' is at least $s' = AD_{AD} - 1 - d - 1 = s - 2$. It then follows that after applying the first k lowest speciations, we have a solution with at least $s - 2k$ more useful speciations, which implies that k can be at least as big as $s/2$ if s is even. If s is odd, k can be as high as $(s - 1)/2$, and there is at least one useful speciation available, hence the lower bound of $\lceil s/2 \rceil$.

□

Proof of Theorem 5: First, we can notice that by including the bridges into M, we ensure that all other added edges are useful speciation edges.

Now, we prove the maximality of the useful matching by induction on $|X \cup Y|$. Given $P = (X, Y, AD_X, AD_Y, B)$, denote by M_P the solution returned by Algorithm 2, and by OPT_P a useful matching of maximum size over instance P.

If $|X \cup Y| = 1$, then the theorem trivially holds, since each useful matching of P contains no edge. Assume the theorem holds for $|X \cup Y| = k$, we show that it holds for $|X \cup Y| = k + 1$.

Let $\alpha \in X \cup Y$ be the last vertex added to D by Algorithm 2, and assume w.l.o.g that $\alpha \in X$. Write $X' = X \setminus \{\alpha\}$, and P' the instance obtained from P by removing α. By induction, since $|X' \cup Y| = k$, $|M_{P'}| = |OPT_{P'}|$. Moreover, by construction, $M_{P'}$ is exactly M_P minus the edge of M_P incident to α, if any.

Assume that α is incident to an edge of M_P. It holds that $|M_P| = |M_{P'}| + 1 = |OPT_{P'}| + 1$. On the other hand, remove from OPT_P the edge incident to α, if any. Then the edges left in OPT_P form a useful matching of P', and thus $|OPT_{P'}| \geq |OPT_P| - 1$. As it has been shown that $|M_P| = |OPT_{P'}| + 1$, it follows that $|M_P| \geq |OPT_P|$, and thus M_P is a useful matching of P of maximum size.

Now, assume that α is not incident to an edge of M_P. Denote by $c(\alpha)$ the connected component of $\mathcal{G}_{P,M_P}$ that contains α.

Claim i. Each vertex β in $Y \setminus c(\alpha)$ is incident to an edge in M_P.

If the claim was wrong, the algorithm would have added an edge between α and β. If, in addition, each vertex of $Y \cap c(\alpha)$ is incident

to an edge of M_P, then each vertex of Y is incident to an edge of M_P, implying that M_P is of maximum size, which completes the proof. Hence assume that there exists at least one vertex β of $Y \cap c(\alpha)$ such that β is not incident to any edge of M_P.

Claim ii. Each vertex γ in $X \setminus c(\alpha)$ must be incident to an edge of M_P(*statement ii*).

Again, the proof is immediate: if the claim was wrong, the algorithm would have defined an edge from β to γ.

Now, consider the set $AD_{\setminus\alpha} = AD_{X\setminus c(\alpha)} \cup AD_{Y\setminus c(\alpha)}$ of AD-components on the sets of vertices $(X \setminus c(\alpha)) \cup (Y \setminus c(\alpha))$. By definition of useful speciation edges, the graph defined by the vertex set $AD_{\setminus\alpha}$ and the edge set containing one edge for each pair $(AD_{X_i} \in AD_{X\setminus c(\alpha)}, AD_{Y_j} \in AD_{Y\setminus c(\alpha)})$ of linked components has no cycles, and thus at least one vertex (AD-component) of degree less than 2. Each such AD-component reduces to a single vertex as otherwise there would be a vertex of this AD-component not incident to any edge of M_P, which is in contradiction with Claim ii. Hence, as α is the last vertex added to D and the algorithm proceeds in decreasing order of AD-component cardinality, the AD-component containing α in X should be of cardinality one, meaning that x is an isolated vertex. Hence $Y \cap c(\alpha) = \emptyset$, and with Claim i it follows that each vertex of Y is adjacent to an edge of M_P, and thus M_P has maximum size. □

LEMMA 3. *Let $P = (X, Y, AD_X, AD_Y, B)$ and $P' = (X', Y', AD_{X'}, AD_{Y'}, B')$ be two instances such that $|X'| = |X|, |Y'| = |Y|, |AD_{X'}| = |AD_X|, |AD_{Y'}| = |AD_Y|$ and $|B| = |B'|$. Then P and P' admit maximum useful matchings of the same size.*

Proof of Lemma 3: Consider two maximum useful matchings M, M' of P, P' respectively and the induced graphs $\mathcal{G}_{P,M}, \mathcal{G}_{P',M'}$. Assume w.l.o.g. that $|M| > |M'|$.

• *Claim (i)*: Since $|X'| = |X|$, $|Y'| = |Y|$ and $|AD_{X'}| = |AD_X|$, it follows that $\mathcal{G}_{P,M}$ contains strictly less connected components than $\mathcal{G}_{P',M'}$.

• *Claim (ii)* Since $|M| > |M'|$, it follows that there exists a node x of X' and a node y of Y' that are not incident to an edge of M'. Then one of the two following cases hold.

Case 1.: x and y belong to different components of $\mathcal{G}_{P',M'}$. Then it holds that M' is not a maximum useful matching, since we can add edge $\{x, y\}$ to M', thus contradicting the assumption that M' is a maximum useful matching of P'.

Case 2.: x and y belong to the same connected component $c(x)$ of $\mathcal{G}_{P',M'}$. We show that we can compute a useful matching M^* of P', such that$|M^*(P')| > |M(P')|$. First, we show that there exist two nodes $x_1 \in X'$ and $y_1 \in Y'$ that belong to a connected component of $\mathcal{G}_{P',M'}$ different from $c(x)$ such that $\{x_1, y_1\}$ is an edge of M'. Notice that if x_1 and y_1 do not exist, then one of the following two cases holds: (2.1) There exists a single connected component in $\mathcal{G}_{P',M'}$, but this violates *Claim (i)*; (2) Each connected component of $\mathcal{G}_{P',M'}$ different from $c(x)$ contains only bridges, which implies that there exist two nodes of $\mathcal{G}_{P',M'}$ (one of x, y and a node that belongs to $(AD_{X'} \cup AD_{Y'}) \setminus c(x)$) not incident to an edge of M' and belonging to different components of $\mathcal{G}_{P',M'}$. But then we fall in *Case 1.* and M' is not a maximum useful matching of P'. Thus nodes x_1 and y_1 exist, so we can compute a useful matching M^* of P' starting from M' as follows: remove $\{x_1, y_1\}$ from M' and add edges $\{x, y_1\}$, $\{x_1, y\}$ to M^*. It follows that M^* is a useful matching for P' with $|M^*| > |M'|$, contradicting the assumption that M' is a maximum useful matching of P'. □

LEMMA 4. *Let $P = (X, Y, AD_X, AD_Y, B = \emptyset)$ and $P' = (X', Y, AD'_X, AD_Y, B' = \emptyset)$ be two instances such that $|X'| - |X| = |AD'_X| - |AD_X|$ with $|X'| \geq |X|$. If P' admits a useful matching M', then P admits a useful matching M such that $|M| \geq |M'| - (|X'| - |X|)$.*

Proof of Lemma 4: Let x_1, x_2 be two nodes of X' in two distinct components of AD'_X. If we join the trees corresponding to x_1 and x_2, leading to a single node $x_{1,2}$, we create a new instance $P^* = (X^*, Y, AD^*_X, AD_Y, \emptyset)$, in which $|X^*| = |X'| - 1$ and $|AD^*_X| = |AD'_X| - 1$. If x_1 and x_2 are incident to edges in M', say $\{x_1, y\}$ and $\{x_2, z\}$, then $M^* = M' \setminus \{\{x_1, y\}, \{x_2, z\}\} \cup \{x_{1,2}, z\}$ is a useful matching for P^*. Otherwise, if x_1 or x_2 is not incident to an edge of M', then construct a matching M^* of P^* from M' by removing the edge incident to x_1 or x_2, if any. In all cases, $|M^*| \geq |M'|-1$. By applying such join operation $|X'|-|X|$ times, we obtain an instance P^* with $|X|$ nodes and $|AD_X|$ components and a useful matching M^* verifying $|M^*| \geq |M'| - (|X'| - |X|)$. By Lemma 3, it follows that P admits a useful matching M of the same size, which concludes the proof. □

Proof of Theorem 6: Let $\mathcal{F}$ be the input forest of Algorithm 1. Let $\mathcal{F}(s)$ be the subset of $\mathcal{F}$ containing the trees G such that $s(G)$ is s or one of its descendants. Let n_s be the total number of useful speciations performed on trees of $\mathcal{F}(s)$ at step s of the algorithm, i.e., after considering node s. We show by induction on the height of s that n_s is the maximum number of useful speciations that can be chosen on the trees of $\mathcal{F}(s)$, which proves the theorem as s can be the root of $\mathcal{S}$. This is trivially true if s is a leaf. So let s be an internal node of $\mathcal{S}$ with children x and y. Let $P = (X, Y, AD_X, AD_Y, B = \emptyset)$ be the instance corresponding to s. Let $|M_P|$ be the number of useful speciations performed by the algorithm for P. Then $n_s = |M_P| + n_x + n_y$.

Suppose we can make another choice of n'_x and n'_y useful speciations on $\mathcal{F}(x)$ and $\mathcal{F}(y)$ respectively, yielding a different instance $P' = (X', Y', AD'_X, AD'_Y, B' = \emptyset)$ for s. Suppose also that P' admits $|M_{P'}|$ useful speciations such that $n'_s = |M_{P'}| + n'_x + n'_y > |M_P| + n_x + n_y = n_s$. Note that by induction, $n_x \geq n'_x$ and $n_y \geq n'_y$, and thus we should have $|M_{P'}| > |M_P|$. Any of the n'_x speciations has the effect of merging two nodes potentially in X, and merging two components potentially in AD_X. Now $|X| = |\mathcal{F}(x)| - n_x$. If $|AD_R|$ is the number of AD-components of R before any speciation, then $|AD_X| = |AD_R| - n_x$. Similarly $|X'| = |\mathcal{F}(x)| - n'_x$ and $|AD'_X| = |AD_R| - n'_x$. This leads to $n_x - n'_x = |X'| - |X| = |AD'_X| - |AD_X|$. In the same manner, $n_y - n'_y = |Y'| - |Y| = |AD'_Y| - |AD_Y|$. From this, $n'_s > n_s \Rightarrow n'_s - n_s > n_x - n'_x + n_y - n'_y = |X'| - |X| + |Y'| - |Y|$. But we cal also deduce from Lemma 4 that $n'_s - n_s \leq |X'| - |X| + |Y'| - |Y|$: a contradiction. □

Proof of Corollary 1: A bridge is created between two AD-components AD_X, AD_Y if and only if there exist two vertices $x \in AD_X$ and $y \in AD_Y$ such that $\{x, y\}$ is an S edge in R and x and y belong to the same AD-component in R. It follows that for a pair

$(\mathcal{F}, \mathcal{S})$ leading to a graph R where AD-components are free from S edges, we are guaranteed that for every node s of S the instance P corresponding to s has no bridges. It follows from Theorem 6 that Algorithm 1 finds a maximum set M of useful speciations, i.e., a set of useful speciations leading to the minimum number ad of AD-components, and to a refinement H with $ad - 1$ NADs. Suppose H is not optimal, i.e., there is an H' with $ad' < ad - 1$ NADs. By Lemma 1, we can assume that the join sequences J' leading to H' has all M' joins of type S first, followed by AD and NAD joins. As M is a maximum set of useful speciations, we have $|M'| \leq M$. If $M' < M$, then after applying the M' speciations, the graph is left with $ad' > ad - 1$ AD-components, requiring more than $ad - 1$ NADs, contradicting the hypothesis. Therefore H is a solution to theMinNADref problem. □