APPENDIX 1: Description of the UMP algorithm used to combine all MP trees into a single graph.

Input file: nexus file (Maddison et al., 1997) with trees, including branch length information, in the Newick Standard tree format (evolution.genetics.washington.edu/phylip/ newicktree.html).

Output file generated by the algorithm: a text file with a list of connections (among all missing node and sampled haplotypes) defining a "UMP graph".

Definitions:

Node haplotype: haplotype connected to three or more neighbors in a graph.

Branch haplotype: haplotype connected to exactly two neighbors in a graph.

Tip haplotype: haplotype connected to a single neighbor in a graph.

Missing haplotype: haplotype shown in a graph but not present in the input data set for the maximum parsimony analysis.

Sampled haplotype: haplotype present in the input data set for the maximum parsimony analysis.

Note that *node* and *branch* haplotypes can be *sampled* or *missing*, and that *tip* haplotypes are always *sampled*.

Direct path between haplotypes: a path on the graph that joins two given haplotypes, without passing through a *sampled* haplotype.

The *length* of a direct path is given by the number of mutations separating the starting haplotype from the haplotype at the end of the path.

Node and *branch* haplotypes along a *direct path* are ordered, thereby occupying each a defined position.

Let t_i be the i^{th} of the n trees included in the input file.

Let u_v^i be the v^{th} of the m *missing node* haplotypes associated to tree t_i.

Let h_α^i be the α^{th} of the k *sampled* haplotypes associated to tree t_i.

For each pair of *sampled* haplotypes $\{h_\alpha, h_\beta\}$, let $S(h_\alpha, h_\beta)$ be the set of all *direct paths* $S^i(h_\alpha, h_\beta)$, in all trees, for which the path *length* is minimum but greater than 1.

Algorithm:

1. All *missing node* haplotypes are given a unique label.
2. do
   - 2.1. $\forall \{t_i, t_j\}, i \neq j, i \in [1, \ldots, n], j \in [1, \ldots, n]$
     - 2.1.1. $\forall \{u_v^i, u_w^j\}, v \neq w, v \in [1, \ldots, m], w \in [1, \ldots, m]$
       - 2.1.1.1.1. re-labeling of nodes (give identical label to u_v^i and u_w^j if *conditions 1* and *2* are satisfied)

   while (re-labeling of nodes has occurred)
3. randomly walk through $L \equiv \left\{ \{h_\alpha, h_\beta\}, \alpha \neq \beta \right\}$
   - 3.1. if $\left| S(h_\alpha, h_\beta) \right| > 1$
     - 3.1.1. $\forall \{t_i, t_j\}, i \neq j$, $i \in [1, \ldots, n], j \in [1, \ldots, n]$, for which $S^i(h_\alpha, h_\beta)$ and $S^j(h_\alpha, h_\beta) \in S(h_\alpha, h_\beta)$

3.1.1.1. re-labeling of *nodes* and *branch* haplotypes along the path of the $S^i(h_\alpha, h_\beta) \in S(h_\alpha, h_\beta)$ (give identical label to haplotypes in t_i *and* t_j if *conditions 3* and *4* are satisfied)

condition 1: we can find at least two cases of a *direct path* from haplotype u_v^i and u_w^j, to a second haplotype $h_\alpha^{i,j}$ for which path *lengths* are identical in tree *i* and tree *j*.

condition 2: Consider all h_α included in (1) a *direct path* from haplotype u_v^i and u_w^j, to a second haplotype $h_\alpha^{i,j}$, for which path *lengths* are DIFFERENT in tree *i* and tree *j* or (2) a *direct path* from haplotype u_v^i to a second haplotype h_α^i, for which the corresponding *direct path* from u_w^j to a second haplotype h_α^j does not exist. For these h_α, there are less than two instances, for $\alpha \neq \gamma$, where $d(h_\alpha^i, h_\gamma^i) = d(h_\alpha^j, h_\gamma^j)$, with h_γ being a *sampled* haplotype with a *direct path* to u_v^i or u_w^j.

condition 3: the compared haplotypes are at the same position in $S^i(h_\alpha, h_\beta)$ and $S^j(h_\alpha, h_\beta)$.

condition 4: u_w^i (and u_v^j, in the case where two *node* haplotypes are compared) do not exist.

Example:

The input file consists in trees t_1 to t_3 of Figure 3a. It is formatted as follows (PAUP tree output file):

```
#NEXUS
Begin trees;
tree1 = ((A:2,E:3):1,((C:2,D:3):0,(B:3,F:4):1):0);
tree2 = ((A:2,E:3):1,(((C:2,D:3):0,B:4):0,F:4):0);
tree3 = ((A:2,B:3):1,(((C:2,D:3):0,E:4):0,F:4):0);
```

End;

Below, we give examples of testing conditions 1 to 4.

Example of testing *condition 1*:

Comparison of *missing node* haplotypes u_1^1 and u_6^3. We need to consider all the *direct paths* from u_1^1 and u_6^3 to *sampled* haplotypes.

u_1^1	
Sampled haplotype	*Path length*
A	*2*
B	*5*
C	*3*
D	*4*
E	*3*
F	*5*

u_6^3	
Sampled haplotype	*Path length*
A	*2*
B	*3*
C	*3*
D	*4*
E	*5*
F	*5*

There are at least two cases of a *direct path* from haplotype u_1^1 and u_6^3 to a second *sampled* haplotype for which path *lengths* are identical in tree 1 and 3 (there are actually 4 cases, to A, C, D, and F). *Condition 1* is satisfied.

Example of testing *condition 2*:

Lets consider the *direct paths* from haplotype u_1^1 and u_6^3 to a second *sampled* haplotype for which path *lengths* are DIFFERENT in tree 1 and 3: there are two cases, to haplotypes B and E.

So we consider, for the two nodes (u_1^1 and u_6^3), the paths of *sampled* haplotypes B and E to all the other *sampled* haplotypes connected to u_1^1 and u_6^3.

u_1^1	
path	*Path length*
B-A	*7*
B-C	*6*
B-D	*7*
B-E	*8*
B-F	*8*
E-A	*5*
E-B	*8*
E-C	*6*
E-D	*7*
E-F	*8*

u_6^3	
path	*Path length*
B-A	*5*
B-C	*6*
B-D	*7*
B-E	*8*
B-F	*8*
E-A	*7*
E-B	*8*
E-C	*6*
E-D	*7*
E-F	*8*

Because there is more than 1 path with the same *length* (ex: d(B,C) in tree 1 = d(B,C) in tree 3 = 6; d(E,B) in tree 1 = d(E,B) in tree 3 = 8), u_1^1 and u_6^3 are not given the same label.

Example of testing *conditions 3 and 4:*

We compare two *direct paths* between sampled haplotypes B and C, in trees 1 and 2. Each line describes one path, i.e. the haplotypes encountered along the path, with unnamed branch haplotypes represented by dashes:

path in tree 2: C — u_4^2 u_5^2 - - B

path in tree 1: C - u_2^1 - - - B

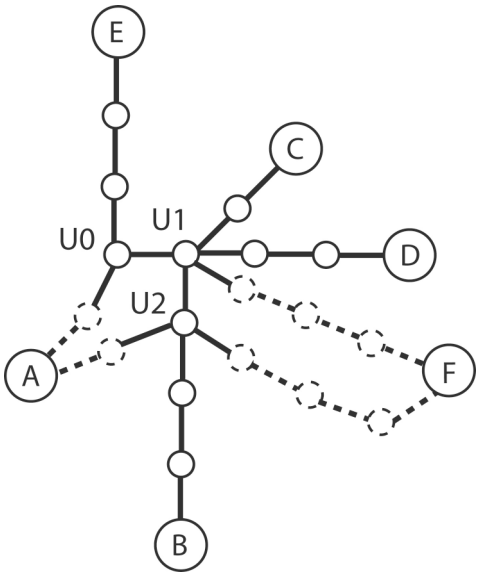
u_4^2 and u_2^1 are at the same position along the path (condition 3). u_4^1 or u_2^2 do not exist (*condition 4*). Therefore, u_4^2 and u_2^1 are given the same label. Similarly, a *branch* haplotype (-) in tree 1 is given the same label as u_5^2. After these two comparisons, the path both in tree 1 and in tree 2 will be updated as follows: C — u_4 u_5 - - B

Here is one possible output of the algorithm*:*

U0 → U1 Dist: 1; U0 → A Dist: 2; U0 → E Dist: 3; U1 → C Dist: 2; U1 → D Dist: 3;

U1 → U2 Dist: 1; U2 → B Dist: 3; U2 → F Dist: 4; U1 → F Dist: 4

This would result in the following graphical representation:



Note that this result is equivalent, but somewhat graphically different from the left union graph presented in Figure 3b. Indeed, connections U1-F and U2-F are partially merged in figure 3b. An additional step (merging branches and branch nodes starting from tip nodes) in our algorithm can lead to this graphical simplification. The 8 branches and 8 branch nodes that would be merged are indicated as dashed in the figure above. However, implementing this additional step would not modify the number of errors or the number of loops calculated for the comparison of maximum parsimony to algorithmic methods presented in this study.