function linkTrees (skeleton $Sk(V_s, E_s)$)

1) let $S := \texttt{root}(Sk)$
2) let $R_S := \{s \in R | \lambda(s) = S\}$
3) for all children S_i of S
   a) let Sk_i be subtree of Sk rooted at S_i
   b) (r_i, c_i) := **linkTrees**(Sk_i)
4) let $\texttt{cost} := \sum_i c_i$
5) for all i, let $l_i := \mu(S, c_i)$
6) for all i, define $p_i \in \{0,1\}^m$ s.t. $p_i[l_i] \neq r_i[l_i]$ and for all $j \neq l_i$, $p_i[j] = r_i[j]$
7) let $\tau := R_S \cup (\cup_i \{p_i\})$
8) let $D \subseteq C$ be the set of characters where taxa in τ differ
9) guess root taxon of S, $r_S \in \{0,1\}^m$ s.t. $\forall i \in C \setminus D, \forall u \in \tau,\ r_S[i] = u[i]$
10) let c_S be the size of the optimal Steiner tree of $\tau \cup \{r_S\}$
11) return $(r_S, \texttt{cost} + c_S)$

Fig. 3. Pseudo-code to construct and link imperfect phylogenies

Function `linkTrees` takes a rooted skeleton Sk (sub-skeleton of PP) as argument and returns a tuple (r, c). The goal of function `linkTrees` is to convert skeleton Sk into a phylogeny for the taxa that reside in Sk by adding edges that mutate M. Notice that using function λ, we know the set of taxa that reside in skeleton Sk. The phylogeny for Sk is built bottom-up by first solving the phylogenies on the sub-skeleton rooted at children super nodes of Sk. Tuple (r, c) returned by function call to $\texttt{linkTrees}(Sk)$ represents the cost c of the optimal phylogeny when the label of the root vertex in the root super node of Sk is r. Let $S = \texttt{root}(Sk)$ represent the root super node of skeleton Sk. R_S is the set of input taxa that map to super node S under function λ. Let its children super nodes be $S_1, S_2, \ldots$. Assume that recursive calls to `linkTrees`(S_i) return (r_i, c_i). Notice that the parents of the set of roots r_i all reside in super node S. The parents of r_i are denoted by p_i and are identical to r_i except in the character that mutates in the edge connecting S_i to S. Set τ is the union of p_i and R_S, and forms the set of