This merges data from the b tree into the a tree; that is, it transforms the a tree so that it accurately represents both the input used to construct a and the input used to construct b.

merge\_tree(a,b):

(This is only called on nodes that are either both roots of their respective trees,

or that represent the same utterance or action. It modifies a and its descendants,

and leaves b unchanged.)

if neither a nor b have children, return

if a and b each have exactly one child:

if the children represent the same utterance or action (in the case of utterances, this is determined with the same\_main\_event() method) then

call merge\_tree on those nodes

otherwise, add b's child (and all its descendants) as an **alternative** to a's child

else if a and b have the same number of children then:

if they have the same children (again, as determined by same\_main\_event())

in the same order, then

merge\_tree each pair of children

if they have the same children in a different order then

update a's child relationship matrix to allow for b's permutation

then merge\_tree each pair of children

else if a and b both either have one child or are marked that their children are alternates then:

add all children of b as alternate children of a

(note: this should be changed to find children that are the same and merge them)

otherwise (the trees have non-identical sets of children):

I have some ideas about how to solve this, but I haven’t fully figured it out yet