16.1 Finishing suffix trees

We have seen that the extended suffix trees we’ve been dealing with so far require $O(n^2)$ space if $n$ is the length of the text. Can we do better? Today we will see how to construct suffix trees in linear space with respect to the length of the text. (To see how to construct suffix trees in linear time, take CS182.) We can transform our extended suffix tree into a more space-efficient suffix tree in two steps.

**Step 1:** Whenever there is a path with no branching in the extended tree, we will replace the entire path with a single edge labeled by the string corresponding to this path. In this new suffix tree, edges are labeled by strings rather than single characters. Of course, rather than building the extended suffix tree and then shortening the branches, we can simply build the more space-efficient tree from the very beginning.

Observe that this new tree is branched at every internal node, i.e. no internal node has just one child. If such a tree has $n$ leaves, then there are no more than $n$ internal nodes. So the number of nodes and the number of edges in the tree are both $O(n)$. The total size of all the strings labeling the edges, however, is $O(n^2)$, so we haven’t quite achieved linear space yet. The second step addresses this.

**Step 2:** We will replace each string $w$ that labels an edge with a pair of positions $(i, j)$ defined by

- $i =$ initial position of $w$ in $x$,
- $j =$ final position of $w$ in $x$.

On each of the $O(n)$ edges, we now have a constant size label $(i, j)$, so the overall space to represent the tree is now $O(n)$ as desired.

To see these ideas in action, consider the two examples from last lecture.

1. Let $x = aaaaa$. The extended suffix tree is...
Note that every internal node is already branched, so step 1 has no effect on this tree. Furthermore, every edge is labeled by a single character, so step 2 will actually increase the space requirements of our tree. Nevertheless the big-O space will remain linear.

2. Now let $x = a_1a_2a_3a_4a_5$ where $a_i \neq a_j$ whenever $i \neq j$. In this case the extended tree is

The first step shortens nonbranching paths, yielding The first step reduces the number of nodes and edges from $O(n^2)$ to $O(n)$, but this tree still requires quadratic space due to the string labels. The second step takes care of this.
The final tree indeed requires only linear space.