## A Supplementary Material

### A.1 Proof of Proposition 1

We will show that, given an arbitrary strictly-ordered d-tree $\mathcal{D}$, we can perform an invertible transformation to turn it into a binary c-tree $\mathcal{C}$; and vice-versa. Let $\mathcal{D}$ be given. We visit each node $h \in \{1, \ldots, L\}$ and split it into $K + 1$ nodes, where $K = |M_h|$, organized as a linked list, as Figure 3 illustrates (this will become the spine of $h$ in the c-tree). For each modifier $m_k \in M_h$ with $m_1 \prec_h \ldots \prec_h m_K$, move the tail of the arc $\langle h, m_k, Z_k \rangle$ to the $(K + 1 - k)$th node of the linked list and assign the label $Z_k$ to this node, letting $h$ be its lexical head. Since the incoming and outgoing arcs of the linked list component are the same as in the original node $h$, the tree structure is preserved. After doing this for every $h$, add the leaves and propagate the yields bottom up. It is straightforward to show that this procedure yields a valid binary c-tree. Since there is no loss of information (the orders $\prec_h$ are implied by the order of the nodes in each spine), this construction can be inverted to recover the original d-tree. Conversely, if we start with a binary c-tree, traverse the spine of each $h$, and attach the modifiers $m_1 \prec_h \ldots \prec_h m_K$ in order, we get a strictly ordered d-tree (also an invertible procedure).

### A.2 Proof of Proposition 3

We need to show that (i) Algorithm 1, when applied to a continuous c-tree $\mathcal{C}$, retrieves a head ordered d-tree $\mathcal{D}$ which is projective and has the nesting property, (ii) vice-versa for Algorithm 2. To see (i), note that the projectiveness of $\mathcal{D}$ is ensured by the well-known result of Gaifman (1965) about the projection of continuous trees. To show that it satisfies the nesting property, note that nodes higher in the spine of a word $h$ are always attached by modifiers farther apart (otherwise edges in $\mathcal{C}$ would cross, which cannot happen for a continuous $\mathcal{C}$). To prove (ii), we use induction. We need to show that every created c-node in Algorithm 2 has a contiguous span as yield. The base case (line 3) is trivial. Therefore, it suffices to show that in line 8, assuming the yields of (the current) $\psi(h)$ and each $\psi(m)$ are contiguous spans, the union of these yields is also contiguous. Consider the node $v$ when these children have been appended (line 9), and choose $m \in \bar{M}_h^j$ arbitrarily. We only need to show that for any $d$ between $h$ and $m$, $d$ belongs to the yield of $v$. Since $\mathcal{D}$ is projective and there is a d-arc between $h$ and $m$, we have that $d$ must descend from $h$. Furthermore, since projective trees cannot have crossing edges, we have that $h$ has a unique child $a$, also between $h$ and $m$, which is an ancestor of $d$ (or $d$ itself). Since $a$ is between $h$ and $m$, from the nesting property, we must have $\langle h, m, \ell \rangle \not\prec_h \langle h, a, \ell' \rangle$ Therefore, since we are processing the modifiers in order, we have that $\psi(a)$ is already a descendent of $v$ after line 9, which implies that the yield of $\psi(a)$ (which must include $d$, since $d$ descends from $a$) must be contained in the yield of $v$.