JOURNAL OF MATHEMATICAL PSYCHOLOGY 25, 206–225 (1982)

Lattice-based Similarity Measures between Ordered Trees

STEPHEN C. HIRTLE

University of Michigan

A clustering algorithm has recently been developed by Reitman and Rueter to express both the structure of chunking in multi-trial free recall and the order of chunk production. The resulting ordered trees differ from ordinary rooted trees in that the elements of a chunk, at any level, may be restricted to a specific ordering. In order to make comparisons of long-term memory structures between subjects, a measure of the similarity between trees is needed. Previously developed similarity measures are shown to be inadequate for ordered trees. Lattice theory is used to generate new similarity measures suited to these richer structures. First, ordered trees are shown to form a nonmodular, graded lattice. Then, moves through this lattice are defined and used to produce several distance measures. These new measures are compared both to each other, and to existing measures, by examining the properties of each measure, and through application to hypothetical trees. The lattice-based measures prove to be theoretically superior, but lack computational ease. The general problem of describing paths in a nonmodular lattice is discussed.

A recently developed clustering technique to describe regularities in free and cued recall generates a data structure known as an ordered tree (Reitman & Rueter, 1980). In introducing ordered trees, Reitman and Rueter left untouched to question of how to compare different ordered trees drawn from the same set of elements. Such comparisons may be useful in contrasting organizations of different classes of subjects, such as experts and novices (McKeithen *et al.*, 1981), or tracing learning within a subject across time. The purpose of this paper is to examine the similarity of ordered trees in detail.

Ordered trees are a new twist on an old theme. Trees, in general, have been used for some time to represent pairwise dissimilarities (Hartigan, 1975; Johnson, 1967). The trees used are of many different types. Two main distinguishing characteristics are whether the tree is *rooted*, resulting in a hierarchical arrangement, or *free*, with no hierarchy implied. A second feature distinguishing trees concerns which nodes are labeled. The most common arrangement is for terminal nodes to be labeled and nonterminal nodes to be unlabeled.

Johnson's (1967) clustering schemes resulted in rooted trees in which only the terminal nodes are labeled. In addition, Johnson assigned ordinal valued heights to intermediate nodes as a indication of the level at which a cluster is formed. Boorman and Olivier (1973) denote such trees as *valued* trees. When data concerning heights is not available due to the clustering technique, the resulting graph is denoted as a *bare* tree.

Not all trees are rooted. Cunningham (1978) discussed free trees as graph-theoretic

0022-2496/82/030206-20$02.00/0

representations of psychological distance. By removing the root, Cunningham does not require hierarchical relationships. Instead, path lengths through the tree represent psychological distances. This formulation has also been discussed by Carroll (1976), and Sattath and Tversky (1977), among others. In addition, free trees allow for both the terminal and nonterminal nodes to be labeled. Cunningham (1978) further extends this representation to account for asymmetry by introducing bidirectional trees, allowing the length of the link from node *a* to *b* to be different from the length of *b* to *a*.

Another type of hierarchical tree is the *PQ*-tree (Booth & Lueker, 1976), which is used to represent classes of permutations that contain consecutive subsequences. *PQ*-trees, while virtually unknown to psychology, are highly relevant to the fields of graph theory and computer science. They are rooted, bare trees which contain two types of nonterminal nodes: *P*-nodes in which the elements are permutable to any ordering, and *Q*-nodes in which the elements are permutable only to two orders, the given order and its inverse.

The Reitman–Rueter (1980) clustering algorithm was created to explain the structure of chunks and the order of chunk production in multi-trial free recall. The algorithm results in an ordered tree which is identical to *PQ* trees, with an additional type of nonterminal node, one where the elements are fixed to a single order. Reitman and Rueter call *P*-nodes nondirectional, *Q*-nodes bidirectional, and the third class of nodes unidirectional. It is crucial to note that neither Reitman–Rueter ordered trees nor *PQ*-trees are based on an underlying similarity matrix of pairwise distances between items, but rather on the regularities of elements within a set of linear strings.

The focus of this article is on the similarity between Reitman–Rueter ordered trees generated from the same recall set. Similarity between ordinary rooted trees (both valued and bare) has been addressed by Boorman and Olivier (1973). Likewise, Cunningham (1980, Note 1) addresses similarity between free trees. However, as will be discussed below, neither of these approaches can adequately be applied to ordered trees.

This paper presents a method for assessing the similarity of ordered trees.[1] The first section of this paper introduces Reitman–Rueter trees and defines some basic terminology. Having set forth some knowledge of Reitman–Rueter trees, section two reviews previous work on similarity between trees, and details why ordered trees are a special case. In section three, lattice theory is introduced to describe formally the relationships among Reitman–Rueter trees in fine detail. The lattice framework suggests several distance measures, which are then compared and contrasted in the final section.

[1] The approach taken here could also be applied to *PQ*-trees with only slight modifications in the height function and covering relationships defined later in this paper. However, I have chosen only to address ordered trees at this time.

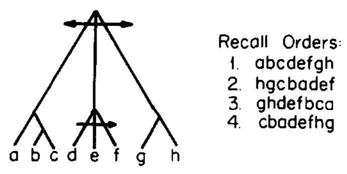


FIG. 1. Tree diagram for $\langle(a(bc))[def](gh)\rangle$.

REITMAN–RUETER ORDERED TREES

The Reitman–Rueter (1980) algorithm is a deterministic clustering algorithm designed to describe the regularities in free and cued recall of a fixed memory set. The fundamental assumptions of the model of recall are that items are organized into chunks, and that the subject recalls chunks as units, recalling all of one chunk before proceeding to the next. The chunks are mentally organized into a hierarchical tree where the terminal nodes of the tree represent the items to be recalled. An assumption is made that the traversal of the structure can be constrained by directionality at any node. Unidirectional chunks can only be accessed in one order, as, for example, reciting the alphabet. Bidirectional chunks can be accessed in a single order or its reverse, as in counting up to or down from ten. Finally, nondirectional chunks can be accessed in any order.

Figure 1 is a sample tree. The single-headed arrow indicates the node is unidirectional; the double-headed arrow indicates bidirectionality. Note that the letters *a–h* are merely placeholders, as the actual items to be recalled may come from any of a variety of domains.

The data for deriving an ordered tree must be a set of complete recall orders, or strings, from a well-learned set of items. From this set of recall strings, the Reitman–Rueter algorithm finds the set of all chunks, and represents this set as an ordered tree. Briefly, the algorithm recursively examines strings "top down" for chunks. As an example, a set of recall orders which would generate the tree in Fig. 1 are shown next to the figure.

Ordered trees can be represented either by a tree diagram with arrows indicating directionality as shown in Fig. 1, or by a parenthetic expression where normal parentheses indicate nondirectionality; angle brackets represent bidirectionality, and square brackets represent unidirectionality. The corresponding expression for Figure 1 is $\langle(a(bc))[def](gh)\rangle$.

Before continuing, some basic terms must be defined. Consider a finite *alphabet* of *items* to be recalled. A *chunk*[2] is an ordered set of *elements* enclosed by a pair of *delimiters*, where each element is itself either a chunk or an item. Delimiters denote

[2] The term chunk refers to explicit chunks, as denoted by the tree structure. However, this is just one of many possible definitions. For example, recall-chunks are later defined as any set of items that is always recalled together, such as *ab* in the tree $\langle abc\rangle$.

one of three types of chunks, as previously noted. The elements of unidirectional chunks $\{[\ ]\}$ are always recalled in the specified linear order. The elements of bidirectional chunks $\{\langle\ \rangle\}$ are recalled in either the specified order or its inverse. The elements of nondirectional chunks $\{(\ )\}$ can be recalled in any order.

Each chunk must have a minimum of two elements. If C is a chunk, $C-$ is the set of items within C, or C without any delimiters. Note that the following equivalencies exist:

(i) $\langle e_1 \cdots e_n\rangle = \langle e_n \cdots e_1\rangle$;

(ii) $(e_1 \cdots e_n) = (p(e_1 \cdots e_n))$, where $p(E)$ is any permutation of the ordering of the elements of E;

(iii) $\langle e_1 e_2\rangle = (e_1 e_2)$;

(iv) $[e_1 e_2 e_3] = [e_1 [e_2 e_3]] = [[e_1 e_2] e_3]$.

A *tree* T is a chunk, as defined recursively above, where $T-$ is the entire alphabet.

The *language* L of a tree is the set of recall orders conguent with a particular tree, that is, orders obtainable by traversal through the tree. The size, or *cardinality*, of the language of a tree T is denoted $n(T)$ and can be computed from the tree structure by the product from all nodes of the number of possible ways the branches of each node can be ordered. Specifically,

$$n(T) = 2^j n_1!\, n_2! \cdots n_k!, \tag{1}$$

where

j = number of bidirectional nodes,[3]

k = number of nondirectional nodes,

n_i = number of branches in the ith nondirectional node, $\quad i = 1,\ldots, k$.

Since $n(T)$ grows exponentially, Reitman and Rueter introduced the $PRO(T) = \log_2 n(T)$, as a more useful measure of the size of a language.[4] The right side of Eq. (1) represents the *factorization* of the tree partitioning T into chunk type and size. For the tree T in Fig. 1, the factorization is 2^4, $n(T) = 16$ and $PRO(T) = 4$.

Ordered trees have an advantage over non-ordered trees in describing mental organizations in that they are based on a theory of recall. However, while it is important to describe the particular organizations, it also may be desirable to compare different organizations. This paper focuses on the latter case, how to compare two separate trees generated over an identical alphabet in order to measure the degree of similarity, or perhaps the amount of change.

[3] For Eq. (1) it will be assumed that $(e_1 e_3)$ is classified as a bidirectional chunk. However, for the calculation of $n(T)$ it is irrelevant, as long as each chunk is uniquely classified.

[4] The logarithm was originally reported to be a natural logarithm (Reitman & Rueter, 1980). More recently, Rueter (Note 2) has favored the use of logarithm base 2 as conceptually consistent with ideas found in information processing.

Previous Distance Measures

Several distance measures have been developed for use with trees, partitions, and sets. However, none of the measures to be described are directly applicable to ordered trees. In this section, I review previous methods and discuss the problems in extending the new measures to ordered trees.

Partitions

Boorman and Olivier (1973) categorize measures into two classes: those based on transforming one tree into another, and those derived from a representative, yet simpler structure, e.g., partitions or incidence matrices. Transformations (least-moves) measures are conceptually simple, yet computationally difficult, while derived measures are comutationally easy but not always conceptually clear.

Combining these two approaches, Boorman and Olivier develop metrics that are based on transformations of partitions. These metrics are computationally straihgtforward, yet offer a least-moves interpretation at the tree level.

A hierarchical tree can be considered an ordered set of partitions. For example, the tree $(((ab)(cd))e)$ is the set of partitions $\{\{abcde\}, \{abcd\ e\}, \{ab\ cd\ e\}, \{a\ b\ c\ d\ e\}\}$. There is a height associated with each partition. If the tree is valued, the heights are given. If the tree is bare, heights can be assigned according to the number of elements in the partition.[5]

Therefore, given two trees, there exists exactly one partition at every height for each tree. The problem of calculating a distance between two trees reduces to calculating the distance between partitions, then summing across all heights. Distances between partitions are relatively easy to calculate (Arabie & Boorman, 1973; Boorman & Arabie, 1972; Day, 1981).

Unfortunately, partitions cannot be easily adapted to the notion of directionality in ordered trees. Consider the tree $T = [abcd]$. To represent T as a series of partitions is a complex problem. One possibility is to ignore directionality and proceed as above. Unfortunately, this tactic does not distinguish between $[abcd]$, $\langle abcd \rangle$, and $(abcd)$, even though conceptually these three trees are quite different. Because information is lost with this approach, an extremely organized memory structure, $[abcd]$, is equated with the complete chaos of a bush, $(abcd)$.

Another approach to defining partitions retains some order information. Partitions on ordered trees can be defined as the set of recall-chunks, where a recall-chunk is defined as any set of items that is always recalled together. For example, the bidirectional chunk $\langle abcd \rangle$ has six nontrivial recall-chunks; *abcd*, *abc*, *bcd*, *ab*, *bc* and *cd*. Thus, recall-chunks distinguish between directional and nondirectional chunks, information that is lost in the previous method. However, no distinction is made between uni- and bidirectionality, e.g., $[abcd]$ has the same recall-chunks as $\langle abcd \rangle$.

A second problem with recall-chunks is computational: each height yields several

[5] A simple rule is the height $= N - p + 1$, where p is the number of elements in the partition. For the example set of partitions the corresponding heights would be 5, 4, 3 and 1.

alternative partitions. For example, tree $[abcd]$ can be partitioned into three sets by one of three possibilities: $\{ab\ c\ d\}$, $\{a\ bc\ d\}$, or $\{a\ b\ cd\}$. To incorporate the Boorman and Olivier partition metric, one would presumably average over each possible partition. However, simply comparing partitions at the lowest nontrivial level of two undirectional trees of length n would require $(n-1)^2$ comparisons. The combinatorial problems of this approach alone make it undesirable. But in addition, the Boorman and Olivier approach has lost its conceptual basis when applied to a unique structure such as ordered trees. The idea of moves is gone.

Free Tree Transformations

Cunningham (1980, Note 1) interprets distance from one free tree to another as the number of steps to transform one tree into another. Transformations occur by either removing or adding a node and branch.

Again, this notion is not directly applicable to ordered trees. There is no method for transforming trees simply by deletion or addition of branches and nodes. Consider two trees, $R = \langle abcd \rangle$ and $BUSH = (abcd)$. One might be tempted to state R and $BUSH$ are one transformation away, as R can be transformed into $BUSH$ by the removal of a bidirectional arrow over the root node. By the same logic, one is then left in the uncomfortable position of calling tree $S = ((abc)d)$ at least two steps (if not more) away from $\langle abcd \rangle$. Yet by looking at the languages of each tree, we find $L(\langle abcd \rangle) = \{abcd, dcba\}$, $L((abcd)) =$ all 24 possible permutations, and $L(((abc)d)) = 12$ orders, all permutations of abc with d either first or last. Surely, as the $L(R) \subset L(S) \subset L(BUSH)$, R should be closer to the S than to $BUSH$.

In summary, while the idea of tree transformations in both the Cunningham, and Boorman and Olivier approach is acceptable, the notion that it is computed as the "fewest moves of the pen to rewrite a tree" needs modification. This issue will be addressed again later in the paper.

Recall-chunks

As an alternative to transformation-based metrics, one could measure similarity based on other summary statistics more closely related to the tree itself. Recall-chunks, for example, summarize an aspect of the tree. A distance measure could be defined as the proportion of recall-chunks in common between two trees. While a metric, this measure is biased by the total number of recall-chunks. Data on 231 pairs of trees from McKeithen *et al.* (1981) indicated that trees with low *PROs* (many chunks) are distant from all other trees. The logarithmic transformation used by McKeithen *et al.* removed this bias; however, the new distance measure is no longer a metric. In addition, as recall-chunks do not distinguish bi- and undirectional strings, undirectionality is underemphasized by this measure.

Languages

As a fourth approach, one could consider the similarity of the languages of two trees as the aspect of comparison. Similarity might be based on the number of orders common to both languages, with an appropriate normalization. Unfortunately, the number of possible recall orders also grows exponentially as the tree becomes bushier, resulting in a problem similar to that with proportion of chunks. Here, however, large, unstructured trees (with consequently large languages) are now artifactually distant.

There is a second problem with looking at similarity of languages. Consider $S = [\langle[ab]cd\rangle\langle efgh\rangle]$ and $T = [\langle[ba]cd\rangle\langle efgh\rangle]$. Intuitively, these two highly-structured trees are nearly identical; only the "*ab*" ordering is reversed. Yet they have *no* recall orders in common. Unlike the reall-chunk measure, a language-based measure places extreme emphasis on similar directionality, resulting in no similarity between trees that are intuitively close.

Tree Distances

Finally, it is possible to compare matrices of tree distances. The most common, but often criticized, measure is the cophenetic correlation coefficient (Farris, 1969; Holgersson, 1978; Sokal & Rohlf, 1962). The cophenetic correlation measures inter-item distances in one tree, and compares those to distances in the second tree, using the ordinary product-moment correlation. In adapting the cophenetic correlation to ordered trees, several problems arise. First, the cophenetic correlation is generally applied to valued trees, where each node is associated with a height derived from the specific clustering algorithm. If not valued, the tree is usually binary where each cluster is the combination of exactly two other clusters. Reitman–Rueter trees are neither valued nor binary, making it difficult to assign inter-item distances.

Furthermore, it is not clear how distances should be assigned to elements within a directional chunk. For example, in the tree $([abcd]e)$, e could be equidistant from a and b, further from b than a, or closer to b than a. Within unidirectional chunks some distances may not be defined, e.g., in $[abcd]$ is there a distance from b to a?

Most tree-building cluster algorithms are distance-based: A matrix of inter-item distances is transformed into a representation (a tree) which most closely fits (in some sense) the original data. Because the Reitman–Rueter algorithm is not built from inter-item distances, later abstraction of distances from the tree is, at best, crude and unrepresentative. Owing to the theoretical uncertainties involved in defining an adequate distance metric appropriate to ordered non-binary bare trees, tree distance measures are considered inadequate.

In summary, methods that are designed for non-ordered hierarchies are not directly transferable to ordered trees. Directionality is ignored in recall-chunk measures, over-compensated for in language-based measures, and undefined for partitions and cophenetic correlations.

What does remain viable from the previous work is the notion of transformations. Both Boorman and Olivier (1973), and Cunningham (1980, Note 1) use the number of steps to transform one tree into another as a measure of distance. However, in order to define what constitutes a unit transformation in an ordered tree, a new approach is needed. The next section introduces lattice theory as an appropriate framework in which to describe transformations.

Lattice of Ordered Trees

In order to define transformations, a framework is needed which encompasses the the structure of ordered trees. Lattice theory provides such a framework, since a lattice succinctly captures the relationship of Reitman–Rueter trees to each other. This section describes the lattice of ordered trees and provides a basis that leads to several more adequate measures of distance between orered trees based on transformations.

As described earlier, every tree defines a language, or set, of recall orders. Some trees are comparable in that they are defined over the same alphabet, and that the language of one is contained in the language of another. The collection of all possible trees for a given alphabet is partially ordered by set inclusion over their languages. For example, $\langle abcd \rangle < ((abc)d) < (abcd)$, as $L(\langle abcd \rangle) \subset L(((abc)d)) \subset L((abcd))$. However, $\langle abcd \rangle$ is incomparable to $[ab\langle cd \rangle]$, as neither language contains the other.

Furthermore, Rueter (Note 2) has shown that the each pair of trees S, T in the partially ordered set has a least upper bound or *join*, denoted $S \vee T$, and each pair has a greatest lower bound or *meet*, denoted $S \wedge T$. The set of trees, then, forms a lattice (Birkhoff, 1967). Figure 2 is the lattice of all trees for a 3-item alphabet. The empty set is appended to represent a *zero* or universally least lattice-element, while nondirectional (abc) represents a *unity* or universally greatest lattice-element. The unity tree is also called the bush, as it contains all possible recall orders.

The meet of two trees T_1 and T_2 is the tree corresponding to the intersection of their respective languages. The join is represented by the closure of, or smallest language containing, the union of languages. Note that the intersection of two

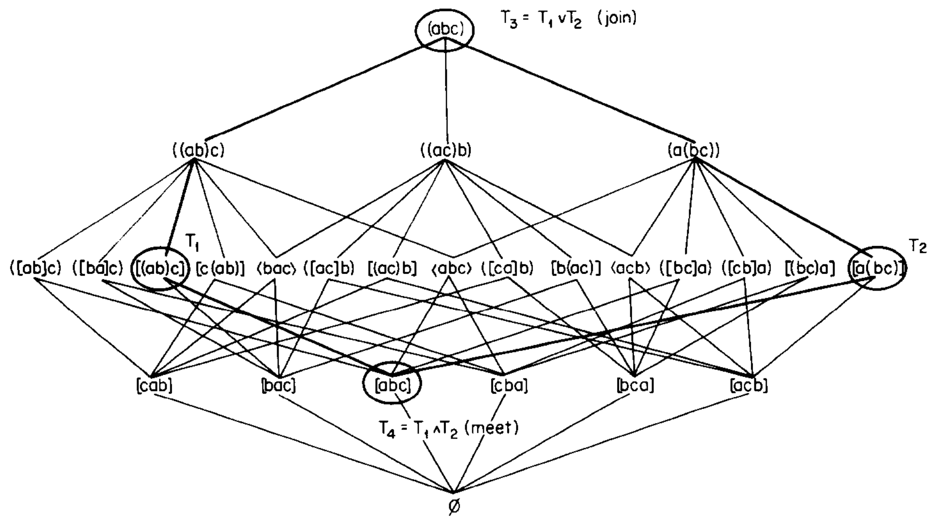


Fig. 2. Tree lattice for a three-item alphabet.

languages is always itself another language, while the union may require the addition of orders to be closed (Rueter, Note 2). Rueter has further shown closure to be a well-defined and unique operation. For example, in Fig. 2, $T_1 = [\langle ab \rangle c]$ and $T_2 = [a\langle bc \rangle]$. The respective languages are $L_1 = \{abc, bac\}$, and $L_2 = \{abc, acb\}$. $L_1 \cap L_2 = \{abc\}$, which implies $T_1 \wedge T_2 = [abc]$. $L_1 \cup L_2 = \{abc, bac, acb\}$. The smallest language containing $L_1 \cup L_2$ is $\{abc, bac, acb, bca, cab, cba\}$, which implies $T_1 \vee T_2 = (abc)$.

At this point, a few technical terms need to be introduced to describe the lattice of trees:

One tree is said to *cover* another if it is greater in the partial ordering and there is no tree between them. Referring back to Figure 2, T_1 covers T_4, but T_3 does not cover T_2.

The *height* of a tree T is the length of the shortest path from the zero to T, where the path is constructed by consecutive coverings. If all paths between a tree and the zero are of equal length, then the lattice is *graded.*

The lattice of ordered trees can be shown to be graded. The height of tree T is given by the formula:

$$h(T) = \sum_{i=1}^{k} [2n_i - 3] + j + 1,$$

where

j = number of bidirectional nodes,

k = number of nondirectional nodes,

n_i = number of branches in the ith nondirectional node, as before.

The point here is that the height of a tree can be calculated from the number and types of nodes in the tree, without actually having to construct a path of consecutive coverings from the zero. The ability to calculate the height easily without constructing the path is vital, since the lattice of all trees for more than a few items is enormous.

The proof that the lattice is graded is given in Appendix 1. While the details are long and technical, an important fact emerges from the proof, an explicit definition of coverings. The proof demonstrates that a tree is covered by another if exactly one of its chunks is covered by another chunk. Furthermore, there are exactly two classes of coverings, directional and nondirectional. The coverings are diagrammed in Fig. 3 and described below.

First, a chunk will cover a directional chunk (Fig. 3a–c), if it can be constructed by adding an additional pair of bidirectional delimiters within the chunk. This can be accomplished in two ways: either by altering an outside pair of undirectional delimiters to bidirectional as in Fig. 3a, or by adding an inner pair of bidirectional delimiters subject to the earlier constraint that all chunks contain at least two elements (Fig. 3b, c).

a) $[....] \longrightarrow <.....>$

b) $[....] \longrightarrow [.<.>..]$

c) $<....> \longrightarrow <.<.>..>$

d) $((.).) \longrightarrow (....)$

FIG. 3. Schematic representations of coverings.

If a chunk is nondirectional, it is covered by another chunk constructed by removing an inner pair of nondirectional delimiters (Fig. 3d). Here we must make one qualification. Bidirectional chunks with two elements will be written as nondirectional for the purpose of this transformation. So $\langle\langle ab\rangle c\rangle$ is rewritten as $((ab)c)$, and is therefore covered by (abc).

Having defined chunk coverings above, a tree will be covered by another tree if exactly one of its chunks is covered, and the remaining tree is untouched. For example, the tree $(a(bc)[def])$ has a factorization of $2^1 3!$ and a height of 5. Tree $(a(bc)[d\langle ef\rangle])$ is a cover occurring within directional chunk $[def]$. The new factorization is $2^2 3!$ and the height by the definition of covering is 6. Tree $(abc[def])$ also covers the original tree with the change occurring within the highest level (nondirectional) chunk. The corresponding factorization is 4!, also at a height of 6. Notice that the equivalence of $(e_1 e_2)$ and $\langle e_1 e_2\rangle$ provides a transposition from directional to nondirectional covers.

Factorization

Possible coverings can be succinctly determined through the factorization. A directional covering increases j, the number of bidirectional nodes, by one, leaving the rest of the factorization expression untouched. A nondirectional covering combines two chunks into one, decreasing either k or j or both, and at the same time increasing the value of n_i for some i.

The permissible coverings can be expressed by another lattice; a lattice of factorizations. The lattice of factorizations for $n = 6$ is shown in Fig. 4. Figure 4 shows, for example, that a tree with factorization 2^4 at height 5 can only be covered by trees with factorizations $2^2 3!$ or 2^5, but not 4!, all at height 6. Figure 4 aso classifies transformations as either directional or nondirectional, depending on the type of covering involved.

The lattice of factorizations will always have an upperright diagonal row, called the directional border, where no additional directional expansions are possible. In Fig. 6, there are seven factorizations on the directional border (2^5,..., 6!). It quickly follows that any tree not on the directional border has at least one undirectional chunk or one bidirectional chunk with three or more elements and therefore can be expanded by a directional move. This is not the case for nondirectional moves. Given $[(ab)\,cd(ef)]$ with factorization 2^2, there is no nondirectional move to factorization 3!, even though the factorization lattice shows such a connection. Therefore, the nondirectional connections in the lattice only represent potential moves.

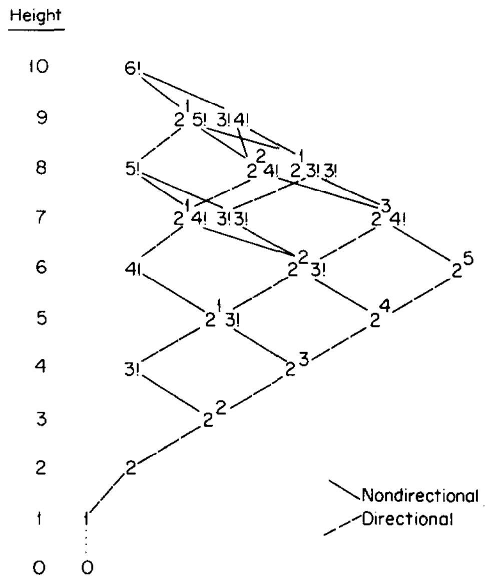


FIG. 4. Lattice of factorizations, $n = 6$.

Free Tree Transformations

If trees are restricted to only contain nondirectional nodes, then the nondirectional move in Fig. 3d is identical to Cunningham's free tree transformation, with one additional caveat. The caveat is a restriction on the root. A rooted tree with n terminal nodes can be considered isomorphic to a free tree with $n + 1$ terminal nodes, where the extra terminal node is appended to the root. If the extra node is not made explicit, rooted trees such as $((ab)c(de))$ and $(((ab)c)de)$ would appear identical as free trees. In sum, free tree transformations can be considered a special case of ordered tree transformations.

LATTICE-BASED DISTANCE MEASURES

The previous section defined the lattice of ordered trees: a graph where each point is a tree, and lines connecting the points correspond to transformations. Transformations were then well defined in terms of the structure of the tree and the factorization. In this section, the distance betwwen two trees is defined as the number of steps of transform one tree into another, or the length of a path connecting two trees in the lattice.

If the trees are comparable in the partial ordering, then as the lattice of trees is graded, the distance between the trees is equal to the difference in heights of the trees. If the trees are not comparable, then the distance depends on the path taken through the lattice.

The first issue to address is which path should be taken through the lattice. Three paths will be discussed: the path from S to T which passes through $S \vee T$ (the upper path), the path through $S \wedge T$ (the lower path), and the shortest path from S to T. The number of steps in each path gives a distance, which will be respectively denoted $U(S, T)$, $L(S, T)$, and $M(S, T)$.

The minimum path is chosen for traditional reasons. The minimum path through any graph is a metric (Harary, 1969) and least-move measures have a natural interpretation. The other paths are important as the join is analogous to the union of two trees (the structure belonging to either tree), while the meet is analogous to the intersection of two trees (the structure belonging to both trees).

Definitions

Next, a precise definition of each distance is needed. Since the lattice is graded, the upper and lower distances can be calculated from the height function. Specifically, given two trees S and T, define the upper distance as the length of the path from S to T through $S \vee T$, or

$$\begin{aligned} U(S, T) &= [h(S \vee T) - h(S)] + [h(S \vee T) - h(T)] \\ &= 2h(S \vee T) - [h(S) + h(T)]. \end{aligned}$$

The lower distance is similarly defined as the path through $S \wedge T$ or

$$L(S, T) = h(S) + h(T) - 2h(S \wedge T). \qquad (2)$$

The minimum distance, $M(S, T)$, is defined as the length of the shortest path from S to T. Unfortunately, $M(S, T)$ can not be calculated from the heights of meets and joins alone; one must find the shortest path by construction, an arduous task.

Modularity

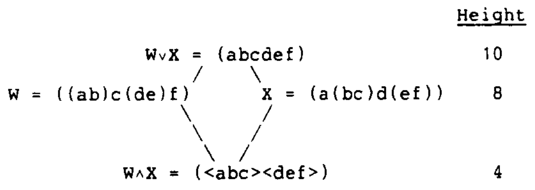
The modularity (Birkhoff, 1967) of the lattice is an important property, as modularity has a direct bearing on path length. Modularity occurs in differing degrees. A finite lattice is upper semimodular if and only if whenever R and S both cover T, then $R \vee S$ covers both R and S. A lattice is lower semimodular if and only if the dual is true, that is, if T covers both R and S, then R and S both cover $R \wedge S$. A lattice is modular if it is both upper and lower semimodular.

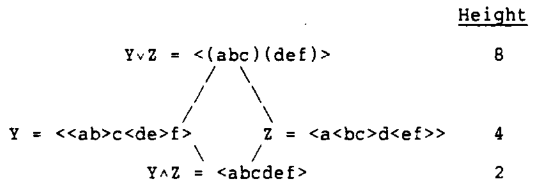
Partitions, discussed in the second section, form a graded upper semimodular lattice. Birkhoff (1967) shows that for upper semimodular lattices $U(S, T) = M(S, T)$, and that for lower semimodular lattices $L(S, T) = M(S, T)$. It is this fact that allows Boorman and Arabie (1972) to define a minimum moves distance measures using Eq. (2) above.

The lattice of ordered trees is not as simple. The lattice of ordered trees is nonmodular, that is, neither upper nor lower semimodular. A counterexample to prove this assertion is given in Appendix 2. As the lattice is nonmodular, it is for each of these paths, the upper, lower, and minimum, to be of different lengths. The different paths result in three different distances. Therefore, closer inspection of each path is needed.

The M-distance is the only measure to have metric properties, which suggests that the M-distance may be the preferred measure. However, as just noted, the M-distance can be difficult to calculate. Several hypothetical examples follow in order to better understand the disadvantages of the U- and L-distance, and at the same time outline the difficulties in calculating the M-distance.

EXAMPLES 1 AND 2. The first two examples below, with trees $W = ((ab)\,c(de)f)$, $X = (a(bc)\,d(ef))$, $Y = \langle\langle ab\rangle\,c\langle de\rangle f\rangle$, and $Z = \langle a\langle bc\rangle\,d\langle ef\rangle\rangle$, indicate the lack of a consistent relationship between the upper and lower path. For W and X, $U(W, X) = 4 < L(W, X) = 8$, while for Y and Z, $U(Y, Z) = 8 > L(Y, Z) = 4$, the exact opposite values.[6]





In terms of transformations, sometimes it maybe shorter to move towards the meet, other times the join. In a sense, to join may be too high or the meet too low. To rely solely on the U-distance or solely on the L-distance would be misleading. In the next example, we see that it is possible for the M-distance to be strictly less than both the upper and lower distances, implying that even the minimum of $U(S, T)$ and $L(S, T)$ is a misleading distance measure.

EXAMPLE 3. The next example demonstrates that the minimum path length may be shorter than both the upper path length and the lower path length. Trees P and Q below are formed in part from Y and Z in the earlier example. In particular, $P = (Yg\langle hij\rangle)$, and $Q = (Z\langle gi\rangle hj)$. Here $U(P, Q) = 7 + 5 = 12$, $L(P, Q) = 8 + 10 = 18$, yet $M(P, Q) = 3 + 2 + 2 + 1 = 8$.

[6] Rueter (Note 2) defines an operation THETA, which interchanges bidirectional and nondirectional delimiters. He further shows THETA preserves, meets and joins, but in a converse manner. This example takes advanatage of the properties of THETA.

```
                                                              height
                 (<(abc)(def)>ghij) = PvQ                       15
                          /
                         / \
                        /   \
                       /     \
                      /       \
                     /         \
       (<<ab>c<de>f>ghij)   (<a<bc>d<ef>>gihj)                  11
            /     \            /  \
           /       \          /  (<a<bc>d<ef>><gi>hj) = Q       10
          /         \        /       /
         /      (<abcdef>ghij)      /                            9
        /                          /
  P = (<<ab>c<de>f>g<hij>)        /                              8
                 \               /
                  \             /
                   \           /
                    \         /
                     \       /
                      \     /
                       \/
                  P^Q = Ø                                        0
```

In order to understand this example it is important to view P and Q as having two separate components: the chunks containing a–f, and the superstructure, or top level node. Within chunk a–f, the meet is closer than the join, but for the superstructure, the meet is zero, much further away from the original chunks than the join. The shortest path is constructed by "meeting" the two chunks containing elements a–f, and "joining" the superstructure. The resulting path can be symbolically written as $(Yg\langle hij\rangle \rightarrow (Yghij) \rightarrow (Y \rightarrow Zghij) \rightarrow (Zgihj) \rightarrow (Z\langle gi\rangle hj)$. The key tree is in the middle. $(Y \rightarrow Zghij)$ is composed of a *partial meet* and a *partial join*. By constructing partial meets and joints, a better approximation of the M-distance can be found.

Meets of Zero

The last example had an additional, unmentioned problem in that the meet was zero. In general, the lower distance falls victim to the same hard restriction that plagued the language-based metric in the last section. It is quite possible for two highly similar trees to have no languages in common, which implies the meet is zero. As the number of items being clustered grows, the lower path becomes arbitrarily long.

This situation can easily occur if one tree has the chunk $[ab]$, and another has $[ba]$. Another common situation generating a zero meet would be if one tree had the chunk $\langle abc\rangle$, while the second has the chunk $\langle bd\rangle$, as in the previous example. One such slip, buried in otherwise identical trees, and the meet falls to zero.[7]

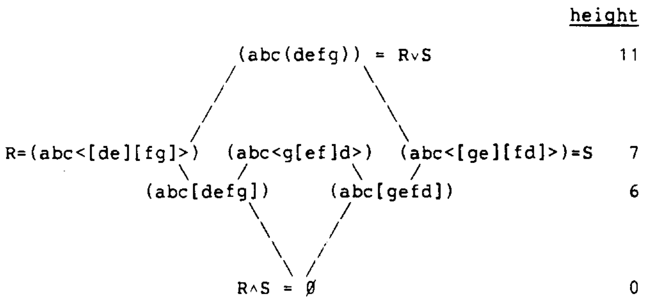
A second problem with lower paths passing through the zero is that they have no interpretation in terms of tree transformations. The lower distance for trees $R = [abcdefg]$ and $S = [bdfagce]$ is 2, yet it would take several moves to transform R into S.

[7] Experimental evidence suggests this is not a trivial point. In analyzing data from McKeithen *et al.* (1981), out of 231 pairs of trees from 22 subjects only 8 pairs had nonzero meets.

The previous distance measures were based on moves through the lattice. All moves has a tree transformation interpretation, except those requiring a move through the zero. If, however we restrict movement to only trees in the lattice, we can retain a transformation interpretation. Therefore, define a new minimum-moves measure $M'(S, T)$ to be the length of the minimum path through the lattice of trees, excluding the zero. Every move through this restricted lattice corresponds to a tree tranformation, unlike moves which pass through the zero of the complete lattice.

To summarize thus far, there are three distances U, L, and M, corresponding to the upper path, the lower path, and the minimum path (through the lattice), and a fourth distance M', corresponding to the minimum path through the restricted lattice. The upper and lower distances are easy to calculate, but provide inconsistent information. In addition, they are not metrics (see Appendix 3 for proof). The minimum distances are metrics (Harary, 1969), but are not easy to calculate. Partial meets and joins provide one means of reducing the estimation, but it is not enough.

EXAMPLE 4. Consider one final example.



Here, the secret in constructing the minimum path is to note the congruency of $\langle[de][fg]\rangle$ and $\langle[ge][fd]\rangle$. Each tree is compatible with *ef*, R with *defg*, and S with *gefd*. This information, unavailable from either the meet or the join, or partial meets or joins, leads to $M'(R, S) = 4$, substantially less than $U(R, S) = 8$, and $L(R, S) = 14$.

One may be tempted to disallow this type of double shift or "piggybacking" in order to remove computational difficulties. However, Boland, Brown, and Day (Note 3) show that metric properties that exist without the piggybacking restriction will not hold with such a restriction. Thus, Example 4 points to both the complexities of computing M and the hazards of imposing arbitrary restrictions.

SUMMARY

Several distance measures for ordered trees have been discussed. Previous distance measures were unable to account for the directionality of ordered trees. The most promising measure was the proportional chunk measure (McKeithen *et al.*, 1981), based on the logarithm of chunks in common. While this measure was easily calculated, the log measure failed in two aspects. It did not distinguish between the two forms of directionality nor was it a metric measure. However, in practice, neither deterrent may prove overly burdensome.

As an alternative, two lattice-based measures, M and M', are shown to match exactly the implicit structure of ordered trees. Each offers a least moves interpretation and satisfies the metric axioms. M includes the zero as a legitimate tree, while M' does not. Unfortunately, a convenient computational method has not yet been formulated. The lattice-based measures were the only measures discussed that could account for directionality.

Ordered tree transformations are an extension of the free tree moves described by Cunningham (1980, Note 1), and as a result, the lattice measures are a generalization of the free tree metrics. Boorman and Olivier (1973) bare tree metrics, tree distance measures, and the proportional chunk measure do not offer least-move interpretations through trees. As a result, their measures are not directly comparable with the lattice measures discussed here.

This paper has explicitly outlined the structure of ordered trees. The trees have been shown to form a graded, nonmodular lattice. The gradation allows a height formula and transformation algorithm to be defined. However, the nonmodularity of the lattice of ordered trees wreaks havoc with defining a distance measure. Nonordered trees form a semimodular lattice, making the problems addressed here unique to ordered (and PQ) trees. What remains to be found is an efficient algorithm for calculating minimum moves metrics in a nonmodular lattice, in general, and for ordered and PQ trees, in particular.

APPENDIX 1

THEOREM. *The lattice of ordered trees is graded.*

Proof. Assume $S > T$. Show either $h(S) = h(T) + 1$, or there exists an R such that $S > R > T$.

Let

$$j_P = \text{number of bidirectional nodes in tree } P,$$

$$k_P = \text{number of nondirectional nodes in tree } P,$$

$$n_{iP} = \text{number of branches in the } i\text{th nondirectional node in tree } P.$$

Define $CH(S)$ as the set of recall-chunks in S. Rueter (Note 2) has shown that if $S > T$, then $CH(S) \subset CH(T)$.

Case 1. Assume $CH(S) = CH(T)$. Define $UCH(T, S)$ as the set of unidirectional chunks in T that are bidirectional in S. Since $S > T$, $UCH(T, S) \neq \varnothing$. Let u be the number of chunks in $UCH(T, S)$, then $j_S = j_T + u$. If $u = 1$, then $h(S) = h(T) + 1$, and we are done. If $u > 1$, then define R as the tree T with only one chunk in $UCH(T, S)$ bidirectional, then $S > R > T$, and we are done.

Case 2. $CH(T) = CH(S) \cup chnk$. Here we assume there is exactly one additional recall-chunk *chnk* in T. We can further assume $UCH(T, S) = \varnothing$. Otherwise, it is trivial to construct R between S and T by latering a chunk in $UCH(T, S)$ to a bidirectional chunk.

Furthermore *chnk* can not be either an implicit recall-chunk or a directional chunk, as it is impossible to remove simply one such chunk and still have a well-formed tree. Furthermore, the parent node must be nondirectional for the same reason, implying the following schemata occurs in S and T:

$$S = \cdots (\qquad\qquad) \cdots$$
$$T = \cdots (\quad (\quad) \quad) \cdots$$

Call the parent node of *chnk* in T *prnt*. Then, since S and T agree everywhere except in *prnt*, the relative heights can be calculated. Let S_{prnt} and T_{prnt} be the trees S and T restricted to *prnt*. Then if *prnt* has n elements, and *chnk* has m elements, the factorization of S_{prnt} is $n!$, while the factorization of T_{prnt} is $m!(n-m+1)!$. Therefore,

$$\begin{aligned} h(S_{prnt})) &= (2n-3)+1 = 2n-2 \\ h(T_{prnt})) &= (2m-3)+(2(n-m+1)-3)+1 \\ &= 2m-3+2n-2m+2-3+1 = 2n-3 \end{aligned}$$

So $h(S) = h(T) + 1$.

Case 3. The third possibility is that $CH(T) = CH(S) \cup CHNK$, where $CHNK$ is a set containing at least two chunks.

First, note that no chunk in CHNK is an explicit nondirectional chunk, for if it was then there exists an R between T and S formed by the union of S and the explicit non-directional.

Let $chnk \in CHNK$. Furthermore, let *prnt* be the smallest explicit chunk in T containing *chnk*.

LEMMA. *chnk is implicit and prnt is directional in T.*

Proof. *chnk* is implicit, otherwise there exists a tree with the chunks of S and the explicit nondirectional chunk *chnk*. If *chnk* is implicit, then by definition *prnt* in T is directional.

The lemma, therefore, implies the following schemata:

$$S = \cdots \langle \quad \langle \quad \rangle \quad \rangle \cdots$$
$$T = \cdots \langle \qquad\qquad \rangle \cdots$$

Note that there is only one additional inner explicit chunk in *prnt* in T, and it is either a bidirectional chunk, or a nondirectional chunk, with exactly two elements. Once again, j is increased by one, implying that the height is increased by one.

Therefore, if $S > T$, then either $h(S) = h(T) + 1$, or there exists an R such that $S > R > T$.

Appendix 2

A finite lattice is upper semimodular if and only if whenever R and S both cover T, then $R \vee S$ covers both R and S. A lattice is lower semimodular if and only if the dual is true, that is, if T covers both R and S, then R and S both cover $R \wedge S$. A lattice is nonmodular if it is neither upper nor lower semimodular (Birkhoff, 1967).

Theorem. *The lattice of languages is nonmodular.*

Proof by contradiction.

Step 1. Show the lattice is not upper semimodular. Consider the sublattice:

```
                                                height
               T = (<ab>cd)                        5
              /            \
      <<<ab>c>d>         <<ab><cd>>                4
            |                |
  R = <<abc>d>           <ab<cd>> = S              3
            \               /
             P = <abcd>                            2
```

Here R covers P and S covers P, but $R \vee S = T$, which does not cover R or S. Therefore, the sublattice is not upper semimodular.

Step 2. Show the lattice is not lower semimodular. Consider the sublattice:

```
                                                height
               T = (abcd)                          6
              /          \
  R = ((abc)d)           (ab(cd)) = S              5
           |                 |
      (((ab)c)d)         ((ab)(cd))                4
            \               /
           P = <(ab)cd>                            3
```

Here T covers R and T covers S, but $R \wedge S = P$, which is not covered by either R or S. Therefore, the sublattice is not lower semimodular and, hence, the entire lattice is nonmodular.

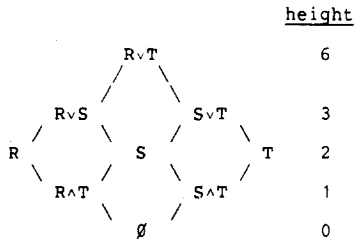
APPENDIX 3

THEOREM. *The U-distance is not a metric.*

Proof. Consider the trees R, S, and T below:

$$\begin{array}{rl|c}
 & & \textit{height} \\
 & R = [ab\langle c[de]\rangle] & 2 \\
 & S = [abc\langle de\rangle] & 2 \\
 & T = [\langle abc\rangle ed] & 2 \\
\text{then} & R \wedge S = [abcde] & 1 \\
 & R \wedge T = [abced] & 1 \\
 & S \wedge T = \varnothing & 0 \\
 & R \vee S = [ab\langle c\langle de\rangle\rangle] & 3 \\
 & R \vee T = (\langle ab\rangle c\langle de\rangle) & 6 \\
 & S \vee T = [\langle abc\rangle\langle de\rangle] & 3
\end{array}$$

which implies the following picture:



Therefore, $U(R, S) = 2$, $U(S, T) = 2$, but $U(R, T) = 8$, which is greater than $U(R, S) + U(S, T)$. Therefore, the U-distance is not a metric.

To show L-distance is not a metric, consider R, S and T above, but where item a is replaced by chunk $(wxyz)$.

ACKNOWLEDGMENTS

This research was supported by NSF Grant BNS-76-82806 to J. S. Reitman at the University of Michigan. The preliminary ideas were presented at the annual meeting of the Society for Mathematical Psychology in Madison, Wisconsin, August 1980.

I would like to than John Jonides, Michael Hacker, and Don Fisher for their helpful comments and encouragement during early stages of the research. I am also grateful to Phipps Arabie and James Cunningham for detailed criticisms of an earlier draft. Finally, I am particularly grateful to Judy Reitman and Henry Rueter for many invaluable discussions and thoughtful criticisms.

Requests for reprints should be sent to Stephen C. Hirtle, Department of Psychology, State University of New York at Albany, 1400 Washington Avenue, Albany, NY 12222.

REFERENCES

BIRKHOFF, G. *Lattice theory* (3rd ed.). Providence, R. I.: American Mathematical Society, 1967.

ARABIE, P., & BOORMAN, S. A. Multidimensional scaling of distance between partitions. *Journal of Mathematical Psychology*, 1973, **10**, 148–203.

BOORMAN, S. A., &ARABIE, P. Structural measures and the method of sorting. In R. N. Shepard, A. K. Romney and S. B. Nerlove (Eds.), *Multidimensional scaling: Theory and applications in the behavioral sciences, Vol. 1.* New York: Seminar Press, 1972.

BOORMAN, S. A. & OLIVIER, D. C. Metrics on spaces of finite trees. *Journal of Mathematical Psychology*, 1973, **10**, 26–59.

BOOTH, K. S., & LUEKER, G. S. Testing for the consecutive ones property, interval graphs and graph planarity using *PQ*-tree algorithms. *Journal of Computer and System Sciences*, 1976, **13**, 335–379.

CARROLL, J. D. Spatial, non-spatial, and hybrid models for scaling. *Psychometrika*, 1976, **41**, 439–463.

CUNNINGHAM, J. P. Free trees and bidirectional trees as representations of psychological distance. *Journal of Mathematical Psychology*, 1978, **17**, 165–188.

CUNNINGHAM, J. P. Trees as memory representations for simple visual patterns. *Memory and Cognition*, 1980, **8**, 593–605.

DAY, W. H. E. The complexity of computing metric distances between partitions. *Mathematical Social Sciences*, 1981, **1**, 269–287.

FARRIS, J. S. On the cophenetic correlation coefficient. *Systematic Zoology*, 1969, **18**, 279–285.

HARARY, F. *Graph theory*. Reading, Mass.: Addison–Wesley, 1969.

HARTIGAN, J. A. *Clustering algorithms*. New York: Wiley, 1975.

HOLGERSSON, M. The limited value of cophenetic correlation as a clustering criterion. *Pattern Recognition*, 1978, **10**, 287–295.

JOHNSON, S. C. Hierarchical clustering schemes. *Psychometrika*, 1967, **32**, 241–254.

MCKEITHEN, K. B., REITMAN, J. S., RUETER, H. R., & HIRTLE, S. C. Knowledge organization and skill differences in computer programers. *Cognitive Psychology*, 1981, **13**, 307–325.

REITMAN, J. S., & RUETER, H. R. Organization revealed by recall orders and confirmed by pauses. *Cognitive Psychology*, 1980, **12**, 554–581.

SATTATH, S., & TVERSKY, A. Additive similarity trees. *Psychometrika*, 1977, **42**, 319–345.

SOKAL, R. R., &ROHLF, F. J. The comparison of dendrograms by objective methods. *Taxon*, 1962, **11**, 33–40.

REFERENCE NOTES

1. CUNNINGHAM, J. P. Similarity between trees: A statistical analysis. Paper presented at the Joint Meeting of the Psychometric Society and the Society for Mathematical Psychology, Hamilton, Ontario, 1978.
2. RUETER, H. R. Thesis, University of Michigan, 1982, in preparation.
3. BOLAND, R. P., BROWN, E. K., & DAY, W. H. Approximating minimum-length-sequence metrics: A cautionary note. Paper presented at the Joint Meeting of the Classification Society and the Psychometric Society, Montreal, Quebec, 1982.

RECEIVED: September 30, 1981; REVISED: June 7, 1982