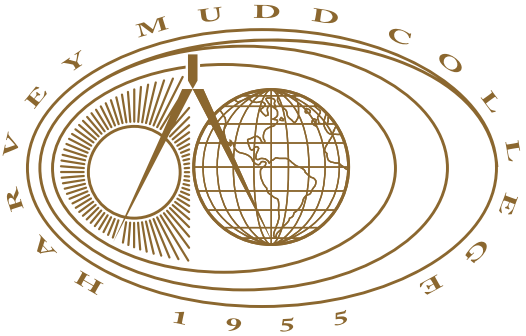
2005

# Constructing Phylogenetic Trees from Subsplits

Akemi Kashiwada
*Harvey Mudd College*




# Constructing Phylogenetic Trees from Subsplits

**Akemi Kashiwada**

---

Professor Francis Edward Su, Advisor

---

Professor Stephen Adolph, Reader

May 6, 2005

HARVEY MUDD
COLLEGE
Department of Mathematics

# Abstract

Phylogenetic trees represent theoretical evolutionary relationships among various species. Mathematically they can be described as weighted binary trees and the leaves represent the taxa being compared. One major problem in mathematical biology is the reconstruction of these trees. We already know that trees on the leaf set $X$ can be uniquely constructed from *splits*, which are bipartitions of $X$. The question I explore in this thesis is whether reconstruction of a tree is possible from *subsplits*, or partial split information. The major result of this work is a constructive algorithm which allows us to determine whether a given set of subsplits will realize a tree and, if so, what the tree looks like.

# Contents

# List of Figures

# Acknowledgments

I would like to thank the members of the Senior Thesis class of 2005 and the Harvey Mudd College Department of Mathematics faculty for their support and helpful feedback on this project.

A special thank you to Professor Francis Edward Su for the invaluable mentoring and advising he has provided me. I would also like to thank Professor Stephen Adolph for his enthusiastic acceptance to be my second reader and Professor Catherine McFadden for allowing me to use one of her phylogenetic trees.

Finally, thank you to Claire Connelly and Professor Melissa O'Neill for providing the hmcthesis class that was used to create this document.

# Chapter 1

# Introduction

Phylogenetic trees represent evolutionary relationships among various taxa. Mathematically they can be described as weighted binary trees and the leaves represent the taxa being compared. When drawn, these trees are usually rooted, that is, there is a node that every species seems to evolve from. However, because the placing of this root is almost arbitrary, we will only be considering unrooted trees in this research.

Although biologists have used phylogenetic trees for some time, the problem of reconstructing these trees is still one of the major issues in mathematical biology (14). There are two main issues with reconstructing trees. First the number of trees increases rapidly with that number of leaves, making it computationally infeasible to try all of them. Secondly, there is a statistical issue with trying to decide which tree is better than another. This thesis will focus on the first problem of minimizing the number of trees that need to be considered as plausible trees.

There already exists several methods for constructing phylogenetic trees, each with its own drawback or limitation. One theoretical method for constructing phylogenetic trees is via *splits* of the leaf set. A *split* of a leaf set $X$ is a pair of nonempty, complementary subsets of $X$.

In 1971, Peter Buneman showed that each split represents the edge of some tree and proved that a labeled tree $T$ is uniquely defined by its splits (8). Since the leaves of phylogenetic trees are taxa, splits reveal which taxa are more closely related; those in the same subset are share a more recent common ancestor than with the taxa in the other subset. So a natural question to ask is whether a tree can be constructed from knowing splits of some of the species. Mathematically we are asking if a given set of splits of subsets of $X$ can construct a phylogenetic tree.

One would imagine that the answer to this question would be some what straightforward, following from Buneman's work. However, as we will demonstrate, there are instances where a tree cannot be reconstructed from subsplits even if Buneman's conditions are satisfied. Focusing on the question of reconstruction from subsplits, I have developed a constructive algorithm that allows us to determine whether a tree can be reconstructed from a set of subsplits and, if so, what this tree looks like and if it is unique.

## 1.1 Literature Review

### 1.1.1 Phylogenetic Trees

Determining phylogenetic trees is one of the fundamental goals of evolutionary biology. There is an ongoing project called the "Tree of Life" where many biologists are working on bringing their knowledge of different parts of the tree together to form one all-encompassing tree (1). Phylogenetic trees are very useful to biologists; they represent a hypothesis about the geneological relationships among organisms derived from data such as DNA sequences. As we can see in Figure 1.1, end points of the tree are the organisms data was taken from and the points where branches diverge are the common ancestor of the resulting organisms. Edges of phylogenetic trees are assigned numbers called *weights* or *branch lengths* that represent some distance between the endpoints of that edge. This distance are generally the number of difference between the species such as the number of places their DNA sequences are different. In our example tree, the numbers associated with each edge are the certainties of the existence of that edge in the actual tree.

The primary use of phylogenetic trees is to summarize the evolutionary relationships among existing organisms. Among its various other uses, phylogenetic trees are used to determine how many times a particular trait has occurred in evolution and to assess when lineages split. Such information is interesting because even if two organisms have similar traits, they did not necessarily evolve from the same ancestor. Such traits are called *homoplasic traits*. There are three major evolutionary processes that lead to homoplasic traits: *convergent evolution*, *parallel evolution* and *evolutionary reversal*. Convergent evolution involves the independent evolution of similar features evolved independently in different species under similar environmental conditions, such as the wing structure of bats and birds. When there is a similar evolutionary trend in distantly related organisms, that is called

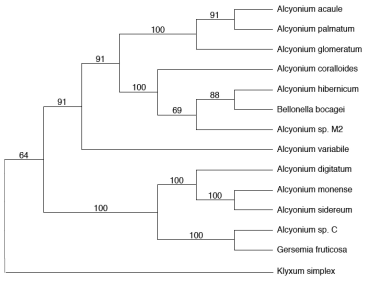



Figure 1.1: Phylogenetic relationship of species in the soft coral genus *Alcyonium* inferred from DNA sequences of the mitochondrial msh1 gene. Tree constructed using maximum parsimony. (19).

parallel evolution. And, evolutionary reversals are those where a trait may revert from a derived state back to an ancestral one. For example, most salamanders have lungs but there are a couple species who have undergone an evolutionary reversal and are lungless.

It is also interesting to note that tree-like representations apply to all evolutionary settings, such as representing the evolution of languages or the copying of a manuscript (7). In fact, the study of these trees is not limited to reconstruction. There papers can be found in economic journals that discuss the diversity represented in evolutionary or phylogenetic trees (see (5), (20), (25), (26)).

For the reasons mentioned above and others, it is very important that phylogenetic (evolutionary) trees can be reconstructed from various types of data.

### Construction Methods

There are three basic categories of methods for constructing phylogenetic trees: *parsimony methods*, *maximum likelihood* and *distance methods*.

The parsimony method requires that several trees be constructed using the same set of organisms but not necessarily the same sources of data (13). Then a new tree is constructed by keeping all of the branchings that appear in all of the given trees.

The maximum likelihood method is usually implemented by a computer program that uses molecular data (14). This method takes into account that mutations arise from changes in the DNA sequence and assume that the frequency of such changes can be estimated independently of other

factors.

There are computer programs such as "Splits Tree" (Huson) and "PHYLIP" (Felsenstein) that take in data and return a phylogenetic tree. Some of these programs accomplish this by searching through a "neighborhood" of an initial tree to find one that best fits the data. However, if one was to run the same data through several different programs based on this construction method, it is likely that they will all return different trees. One of the reasons this is the absence of a standard distance metric on the space of trees, which is extremely complex (2).

A third example of a reconstruction method is called distance methods. Typically, these distances are based on genetic data, such as the number of base pairs that differ for a particular gene. In this method, one is simply given the pairwise distances of the taxa and constructs a possible tree that satisfies these distances. Thus a notion of distance $d(i,j)$ needs to be defined between every pair of species $i$ and $j$. A very helpful tool in determining whether a tree can be reconstructed from a set of pairwise distances is the *four-point condition*.

*Definition* 1.1. A metric $d$ satisfies the *four-point condition* if for any four species $i, j, k, l$, the maximum of

$$\{d(i,j)+d(k,l), d(i,k)+d(j,l), d(i,l)+d(j,k)\}$$

is achieved at least twice.

We can gain some intuition for his definition by considering the following: Suppose $i, j, k, l$ do construct a tree. There is only one unrooted tree topology with four species (see Figure 1.2). Then we define the distance $d$ to be the sum of the weights of the edges that connect $i$ and $j$, i.e.,

$$d(i,j) = \sum_{e \in P} w(e)$$

where $P$ is the set of edges that connect $i$ and $j$ and $w(e)$ is the weight of edge $e$. Then

$$\begin{aligned} d(i,j)+d(k,l) &= (\alpha+\beta)+(\gamma+\delta), \\ d(i,k)+d(j,l) &= (\alpha+\mu+\gamma)+(\beta+\mu+\delta), \\ d(i,l)+d(j,k) &= (\alpha+\mu+\delta)+(\beta+\mu+\gamma). \end{aligned}$$

Since each of the weights is a positive number, the last two sums are the maximum and equal so the four-point condition is satisfied. In fact, we know that the four-point condition is a necessary and sufficient condition for a reconstructing a tree from the pairwise distances.

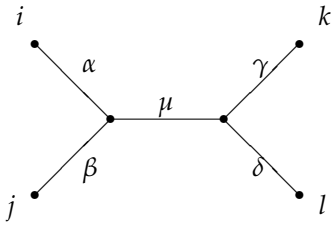



Figure 1.2: Unrooted tree on four species $i, j, k, l$.

**Theorem 1.2.** *(Buneman, 1971) A metric d satisfies the four-point condition if and only if d is realizable by some weighted tree.*

Thus we can use this result to determine whether a set of pairwise distances can be realized as a tree. To see how this method works, see Example 1.3

*Example* 1.3. Let $X = \{A, B, C, D\}$ be our set of species with the distance metric $d(i, j)$ defined as

$$d(A, B) = 2, \;\; d(A, C) = 5, \;\; d(A, D) = 4,$$

$$d(B, C) = 4, \;\; d(B, D) = 3, \;\; d(C, D) = 4.$$

We can easily check to see if these pairwise distances satisfy the four-point condition:

$$d(A, B) + d(C, D) = 6, \;\; d(A, C) + d(B, D) = 8, \;\; d(A, D) + d(B, C) = 8.$$

Since 8 is the maximum and it is achieved twice, the four-point condition is satisfied and we know there exists a weighted (phylogenetic) tree that satisfies these constraints. We can verify that tree in Figure 1.3 satisfies these distance constraints.

### 1.1.2 Splits and Partial Splits

As mentioned, Peter Buneman considered abstractly representing trees in terms of their splits (8). This abstract representation of a tree is useful to consider because it speeds up the reconstruction process through a "divide and conquer" strategy. Given a set of leaves $X$, we can represent a tree by a set of splits $A|B$ where $A, B$ are nonempty, $A \cup B = X$ and $A \cap B = \emptyset$. Each of these splits can be associated to an edge $e$ of the tree by letting $A$ and

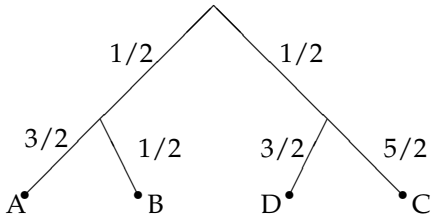



Figure 1.3: One possible phylogenetic tree that satisfies the distance constraints in Example 1.3. (Notice that this tree would be unique if we did not arbitrarily add a root to it.)

$B$ be the leaves of the two subtrees resulting from deleting $e$ from the tree. Since the splits are associated with the edges of the tree, Buneman was able to define the nodes with respect to the splits and showed that this representation was consistent with the definition of graph theoretic trees. Other information about the tree, such as the graph-theoretic *degree* of a node, could also be extracted from the splits. (Recall that the *degree* of a node is the number of edges for which it is an endpoint.) More importantly, Buneman associated a nonnegative number which we think of as the weight of an edge to each split and proved that there exists a unique tree that satisfies these constraints. Thus we know that to every unrooted tree there is associated a unique set of splits which represents the edges of the tree and can reconstruct it. Because the splits are associated with edges, the splits divide the leaves so each subset can be treated as another connected component (with tree structure) justifying the quicker computation.

Working closely with splits, David Bryant wrote an article that considered the properties of neighborhoods of trees using splits with various tree metrics (6). One of the metrics he considers is the *Robinson-Foulds distance*, denoted $d_{RF}$ which is just the number of different splits between two trees. We will discuss this metric further in Section 2.3 where I worked on extending it to trees with different numbers of leaves. He was also a co-author of a paper using splits to construct other types of graphs such as networks (7). This paper was also interesting in that is revealed the broader applications of these techniques to the evolution of languages. One can imagine that these methods, as well as my results, can be used on any type of binary tree where splits of the leaves are known.

Several articles have been written on the reconstruction of trees via its

subtrees and the splits corresponding to these subtrees. One is purely axiomatic, considering the properties one might wish the reconstruction method would satisfy and proving that all desirable properties cannot be simultaneously satisfied (4). The other paper considers reconstruction of a unique tree from four-leaf subsplits (3). Although it seems that my research focusing on reconstruction from $(n-1)$-leaf subsplits could be represented by inducting from the four-leaf case, it is not a trivial matter.

Buneman published another article that gives a constructive algorithm for reconstructing the trees from raw data (9). His method relies on the possibility of finding a rigid circuit graph from the intersection graph of the raw data (e.g. difference in the positions of genome sequence data). From there his algorithm indicated that every node of the tree would represent a clique of the rigid circuit graph and each subtree corresponded to a node of the graph. That is, a node of the graph corresponds to each subtree such that the set of nodes in the subtree correspond to cliques of which the graph node is a member. Because Buneman's algorithm depends on the raw data, there is no clear way to generalize his work to construct trees from subsplits. That reason being is that his algorithm relies on the fact that each node of the intersection graph only corresponds to one of the taxa under consideration. In order to use his construction method on a set of subsplits, each node of the intersection graph would have to be one side of the subsplit so the nodes of the resulting tree would correspond to the sides of the subsplits (which is a collection of the taxa) rather than the individual taxa.

Semple and Steel wrote a paper closely related to my research (22). In their paper they consider *partial partitions* of an $X$-tree which is a tree with all nodes of degree two or less labeled. The focus of their paper is reconstruction from *partial partitions* which is a collection of non intersecting subsets of the labeled nodes that separate them in the tree. These partial partitions can be thought of as generalized subsplits. Using graph theoretical results on chordations of graphs, they prove that a unique phylogenetic tree can be reconstructed from a set of subsplits if the set satisfies three properties. The first of these conditions stipulates that there must exist a tree that satisfy the constraints of the partial partitions. Their work does not suggest any way of knowing if a set of partial partitions satisfies this requirement without checking all possible trees on the given set of labeled nodes. My research precisely answers this question in the case we are given $(n-1)$-leaf subsplits. And, whereas Semple and Steel prove the existence of a unique trees, my work provides a constructive algorithm which allows us to both reconstruct the actual tree and to see the different possible trees if there is

not a unique one.

### 1.1.3 Related Work

Due to the recent surge of integrating mathematics and biology, there are many other investigations of phylogenetic trees involving different disciplines of mathematics. Mike Hendy has been considering the application of Hadamard matrices in computing phylogenies (Hendy). Patrinos and Hakimi's paper provides necessary and sufficient conditions for an $n \times n$ distance matrix to yield an $n$-leaf tree (21). And Bruno Leclerc and Vladimir Makarenkov consider tree metrics and their relationship with other types of trees (18). Although none of these papers are related to my thesis, they provide a sample of the various interests in this field and the many other mathematical applications that can be considered.

## 1.2 Motivation for Thesis

Over summer 2004, I worked on a related project with Professor Francis Su and Professor Claus-Jochen Haake of University of Bielefeld, Germany. We explored the relationship of the *Shapley value* with games induced by phylogenetic trees. Introduced by Lloyd Shapley in 1953, the *Shapley value* of a cooperative game is a well studied concept that is one of the most important in game theory. Shapley showed that this Shapley value is characterized by four axioms that uniquely determine it. Let $N$ be a set of players. Then to every weighted (phylogenetic) tree, we showed that there is an associated cooperative game $v : 2^N \to \mathbb{R}$. We presented a biological interpretation for the Shapley value and proved that the Shapley value of a phylogenetic tree game is characterized by five axioms, analogous to Shapley's earlier theorem. Moreover, we hope that these methods may give new ways to reconstruct a phylogenetic tree from data. I presented a poster on this work at the annual Harvey Mudd College mathematics conference in October 2004 and presented this work in both a Special Session and undergraduate poster session at the 2005 Joint AMS-MAA-SIAM meeting in Atlanta. We are planning to submit our paper for publication in early 2005 so I spent the beginning of the semester refining this paper. (See Appendix A.)

One surprising pattern in our work with the Shapley value on phylogenetic tree games was that every result depended on the size of the partitions in each split. This work led to the question of whether or not another abstract representation of trees would yield another method for reconstruct-

ing trees. Since reconstruction from splits has already been investigated, for my thesis I decided to consider the the possibilities of constructing trees from partial split data; that is, splits of subsets of the leaves. My hope is to combine these *subsplit* results with my Shapley value work to formulate a new method for constructing phylogenetic trees.

# Chapter 2

# Trees and their Splits

Mathematically, phylogenetic trees are binary trees with weighted edges. In this case, the leaves are the species currently in existence, the internal nodes are the ancestor species that current organisms evolved from, and the weights are the measure of distance between the species. In this chapter we will begin with the quick review of graph theoretic terms, then discuss splits.

## 2.1 Graph Theory Terminology

Throughout this thesis we will be considering *n-leaf trees* which are just connected, acyclic graphs with $n$ nodes of degree one called *leaves*. All of the trees we will be working with are *binary* which means that all nodes other than the leaves have degree three. Recall an $n$-leaf tree has $2n - 3$ edges, $n - 3$ of which are internal edges, and that two leaves that are connected by a single node form a *cherry*. See Figure 2.1 for an example of a six-leaf binary tree.

## 2.2 Splits

For the rest of the work discussed we will let $X$ be a set of $n$ leaves.

*Definition* 2.1. A *split* of $X$ is a pair of nonempty unordered complementary subsets $A$ and $B$. We denote such splits by $A|B$. (By unordered we just mean $A|B$ and $B|A$ represent the same split.)

As we noted earlier, every split corresponds to an edge in the tree. So using the tree in Figure 2.1, we see that the splits of this tree are all of the

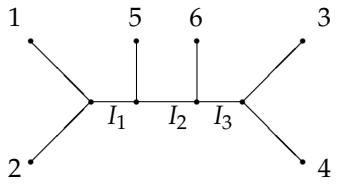



Figure 2.1: Here is a six-leaf binary tree with leaves $1, 2, 3, 4, 5, 6$ and internal edges $I_1, I_2, I_3$. Note that $\{1, 2\}$ form a cherry because they are connected by a single node where as $\{5, 6\}$ is not a cherry.

*trivial splits* defined to be those with only one leaf on one side, i.e., $a|(X \setminus a)$ where $a$ is a single leaf and the *non-trivial* splits:

$$12|3456, \quad 125|346, \quad 1256|34$$

which correspond to $I_1, I_2, I_3$ respectively. The trivial splits correspond to the edges adjacent to the leaves.

*Definition* 2.2. Two splits $A|B$ and $C|D$ are *compatible* if at least one of

$$A \cap C, \ A \cap D, \ B \cap C, \ B \cap D$$

is empty.

**Lemma 2.3.** *Any trivial split $a|(X \setminus a)$ and any other split $C|D$ will be compatible.*

*Proof.* Without loss of generality, let $a \in C$. Then $a \notin D$ so $a \cap D = \emptyset$ thus $a|(X \setminus a)$ is compatible with $C|D$. □

In fact all splits induced by a tree are compatible (8). For example, again consider the splits of the six-leaf tree in Figure 2.1. By Lemma 2.3 we know that the trivial splits are pairwise compatible with all of the splits. For the nontrivial splits, between the first and second split $12 \cap 346 = \emptyset$, the first and third $12 \cap 24 = \emptyset$, and second and third $125 \cap 34 = \emptyset$ so these splits are compatible.

We stated earlier that Buneman proved that trees can be constructed from their splits and by his construction, concluded that this construction was unique. Furthermore, he proved in Theorem 1 that a weighted tree can be uniquely constructed given a set of compatible $2n - 3$ splits (corresponding to the $2n - 3$ edges of an $n$-leaf tree) and a metric on the tree

defined by a weighted sum of the weights of the edges. (8) To see this consider the splits $12|3456, 125|346, 1256|34$ with the six trivial splits. We just showed that these splits were pairwise compatible so we know they construct a tree. However, it is easily verified that it cannot construct any tree other than the one in Figure 2.1.

Now that we know that the set of splits representing all edges defines a tree, we can make a stronger statement.

**Theorem 2.4.** *Any set of $n-3$ compatible nontrivial splits will uniquely construct a binary tree (unrooted).*

*Proof.* From Buneman (8), we know that a tree can be uniquely constructed from $2n-3$ compatible splits that correspond to each of the edges. Since every tree with $n$ leaves will have edges adjacent to the leaves, the corresponding splits of every tree will contain the trivial splits. Thus the trivial splits cannot add any information to the construction of the tree so it must be the case that the tree is uniquely constructed from the $n-3$ splits corresponding to the internal edges. Since those edges have at least two leaves on each side of the split, we know that $n-3$ compatible nontrivial splits will uniquely construct a tree. □

Now we have the necessary background to consider partial split data or *subsplits*.

## 2.3 Robinson-Foulds Metric

As Bryant discusses in (6), the *Robinson-Foulds distance metric* (also called the *partition metric*) can be used to define a neighborhood of a tree. Recall that a neighborhood of a tree is just the set of trees that resemble the given tree within a certain threshold and that these neighborhoods are important for some reconstruction methods discussed earlier. Before we define the Robinson-Foulds metric, we need to define the *symmetric difference* between two sets.

*Definition* 2.5. The *symmetric difference* between two sets $A$ and $B$, denoted $A \ominus B$ is the set of elements belonging to one but not both of $A$ and $B$. That is,

$$A \ominus B = (A \setminus B) \cup (B \setminus A).$$

*Example* 2.6. If $A = \{1,2,3,4\}$ and $B = \{2,4,6\}$ then $A \ominus B = \{1,3,6\}$ since $1,3,6$ are in one set but not both.

Let $\Sigma(T)$ denote the set of splits associated with tree $T$. We define the Robinson-Foulds metric $d_{RF}$ to be

$$d_{RF}(T_1, T_2) = \frac{1}{2}|\Sigma(T_1) \ominus \Sigma(T_2)|.$$

Because we know that these splits correspond to edges of the tree, we can associate a weight to each split. So, for each split $A|B \in \Sigma(T_i)$ for $i = 1, 2$ let $w_i(A|B)$ denote the weight of that split with respect to tree $T_i$. If $A|B \notin \Sigma(T_i)$ then $w_i(A|B) = 0$. Now we can define the weighted Robinson-Foulds distance $d_\omega$ by

$$d_\omega(T_1, T_2) = \sum_{A|B \in \Sigma(T_1) \cup \Sigma(T_2)} |w_1(A|B) - w_2(A|B)|.$$

### 2.3.1 Extending $d_{RF}$

The original Robinson-Foulds metric was only defined for trees with the same leaves. Early in my exploration of splits of a tree, I worked on extending the definition to find the distance between trees with a different number of leaves or different sets of leaves.

The same definition of $d_{RF}$ holds for trees with different numbers or sets of leaves but we need to redefine $d_\omega$. In order to do this extension, we have to define more symbols. Let $T_1$ and $T_2$ be the trees that we want to find the distance between and let the corresponding leaf sets be $L_1$ and $L_2$ where $|L_1| \neq |L_2|$. Let $L = L_1 \cap L_2$, $M_1 = L_1 \setminus L_2$, and $M_2 = L_2 \setminus L_1$. Now define $\Sigma_0(L)$ to be set of all possible splits of $L$ allowing $\emptyset$ to be on one side of the split and define $\Sigma_{00}(L)$ to be the set of ordered splits of $L$ allowing $\emptyset$ to be on one side of the split. Then we define the extended Robinson-Foulds metric, temporarily denoted $\omega_{RFKS}$ for Robinson-Fould-Kashiwada-Su metric, to be

$$\begin{aligned}
&\omega_{RFKS}(T_1, T_2) \\
&\quad = \sum_{A|B \in \Sigma(L)} \Big| \sum_{C|D \in \Sigma_{00}(M_1)} w_1(AC|BD) - \sum_{E|F \in \Sigma_{00}(M_2)} w_2(AE|BF) \Big| \\
&\qquad\qquad + \sum_{C|D \in \Sigma_{00}(M_1)} w_1(C|DL) + \sum_{E|F \in \Sigma_{00}(M_2)} w_1(E|FL).
\end{aligned}$$

Thus we are defining our distance measure to be the sum of (1) the difference in weight of the "subtrees" spanned by $L$ and (2) the weight of the "subtrees" spanned by $M_1$ and $M_2$. (Subtrees is in quotes because they

could include internal edges that are not in a formal subtree.) Because this equation is complicated, it's easier to see what is happening through an example.

*Example* 2.7. Let $T_1$ and $T_2$ be the trees in Figure 2.2 with leaf sets $L_1 = \{1,2,3,4\}$ and $L_2 = \{1,2,7,8\}$, respectively. Also let all weights be 1. Then $L = \{1,2\}$, $M_1 = \{3,4\}$ and $M_2 = \{7,8\}$.

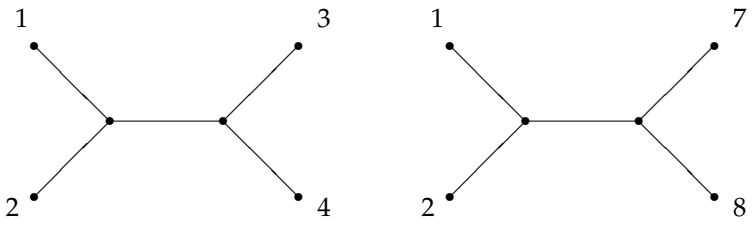



Figure 2.2: The tree on the left is $T_1$ and the tree on the right is $T_2$.

To simplify the computations we are going enumerate all of the splits:

$$\Sigma(L) = 1|2,$$

$$\Sigma_{00}(M_1) = \emptyset|34, 3|4, 4|3, 34|\emptyset,$$
$$\Sigma_{00}(M_2) = \emptyset|78, 7|8, 8|7, 78|\emptyset.$$

Again, to simply the computation we are going to break the equation into parts.

$$\sum_{C|D \in \Sigma_{00}(M_1)} w_1(1C|2D) = w_1(1|234) + w_1(2|134) = 2$$

$$\sum_{C|D \in \Sigma_{00}(M_1)} w_1(C|12D) = w_1(3|124) + w_1(4|123) + w_1(34|12) = 3$$

Similarly, $\sum_{E|F \in \Sigma_{00}(M_2)} w_2(1E|2F) = 2$ and $\sum_{E|F \in \Sigma_{00}(M_2)} w_2(E|12F) = 3$. Therefore $\omega_{RFKS}(T_1, T_2) = |2-2| + 3 + 3 = 6$. This makes sense because we are just calculating the weight of the "subtrees" spanned by $M_1$ and $M_2$ since the subtrees spanned by $L$ are identical.

Notice that when $L = \emptyset$ which means that there are no common leaves between $T_1$ and $T_2$, then

$$\omega_{RFKS}(T_1, T_2) = \sum_{C|D \in \Sigma_{00}(M_1)} w_1(C|D) + \sum_{E|F \in \Sigma_{00}(M_2)} w_1(E|F)$$

which is also reasonable because we just want the distance to be the weights of the trees.

Because we did not see a usefulness for such a distance, we ended our study of the extended Robinson-Foulds metric here and went on to consider reconstructing trees from partial split data.

# Chapter 3

# Subsplits

Again, let $X$ be a set of leaves.

*Definition* 3.1. We define an *m-leaf subsplit* of $X$ to be a split $S = S^L|S^R$ of $Y \subseteq X$ where $|Y| = m$.

Note that the ordering of the subsplit does not matter. That is, $S^L|S^R$ is the same as $S^R|S^L$ but for convenience we call $S^L$ the left-side of $S$ and $S^R$ the right-side of $S$. Using Definition 2.2, we can talk about compatible subsplits.

*Example* 3.2. Consider the following 5-leaf subsplits of a 6-leaf tree

$$123|45, \;\; 456|23, \;\; 234|16.$$

These subsplits are compatible because between the first and second subsplits $123 \cap 456 = \emptyset$, between the first and third $45 \cap 16 = \emptyset$, and between the second and third $23 \cap 16 = \emptyset$.

It is interesting to note that although a set of compatible splits will reconstruct a tree, not every set of compatible subsplits realizes a tree. I claim that the following set of subsplits do not realize a tree.

*Counter Example* 3.3. For the same set of 6 leaves, consider the subsplits

$$12|345, \;\; 16|234, \;\; 15|346.$$

These subsplits are compatible because between the first and second subsplits $345 \cap 16 = \emptyset$, between the first and third $345 \cap 15 = \emptyset$, and between the second and third $234 \cap 15 = \emptyset$. In Section 3.5 we will see why these subsplits do not realize a tree.

Similar to the case with splits, a subsplit indicates the existence of an edge that separates the leaves on each side of the subsplit.

*Definition* 3.4. A tree $T$ *displays* a set of subsplits if for each subsplit $A|B$ there exists an edge $e$ of $T$ such that $A$ are $B$ are subsets of the leaves in the separate components of $T - e$.

Unlike the case with splits, this edge is not unique. For example, in Figure 2.1, given the subsplit 12|46, the edges $I_2$ and $I_2$ both separate 12 and 46. It is also true that two distinct subsplits can correspond to the same edge. Again looking at Figure 2.1 the subsplits 12|56 and 12|35 both correspond to $I_1$.

The first half of this chapter presents some helpful results when considering reconstruction of a tree from nontrivial four-leaf subsplits. Some four-leaf subsplits of the tree in Figure 2.1 are

$$12|56, \quad 15|34, \quad 12|46.$$

I formulated several conjectures about the uniqueness of a tree being related to the number of empty intersections between subsplits and the number of different halves of subsplits all of which were unfruitful. In the following sections I document some results and counterexamples that helped develop some intuition for the problems arising from considering reconstruction from subsplits. In Section 3.5 we begin considering only $(n-1)$-leaf subsplits and present our main result in Section 3.5.1.

## 3.1 Compatibility of Four-leaf Subsplits

The following two lemmas are helpful observations related to the compatibility of subsplits.

**Lemma 3.5.** *Two different four-leaf subsplits $A|B$ and $C|D$ are compatible if and only if $(A \cup B) \ominus (C \cup D) \neq \emptyset$.*

*Proof.* Let $A|B$ and $C|D$ are compatible. Suppose $A \cap C = \emptyset$ while all other intersections are nonempty. Without loss of generality, we can say that forces one element of $B$ to be in $C$ and the other in $D$ and one element of $A$ to be in $D$. However, the other element of $A$, call it $x$, cannot be in $C$; otherwise the intersection would be nonempty. Thus $x \notin C \cup D$ so $(A \cup B) \ominus (C \cup D) \neq \emptyset$. If more than one intersection is empty, then clearly there must be elements in $A|B$ that are not in $C|D$.

Conversely, suppose $(A \cup B) \ominus (C \cup D) \neq \emptyset$. Let $x \in (A \cup B) \ominus (C \cup D)$. Then without loss of generality, let $A = xy$. Then either $A \cap C$ or $A \cap D$ is empty. If $A \cap C$ is empty, then we are done. If it is nonempty that means $y \in C$ since $x \in (A \cup B) \ominus (C \cup D)$. But if $y \in C$ then $y \notin D$ so $A \cap D = \emptyset$. Therefore, $A|B$ and $C|D$ are compatible. □

Another way of stating this result is two four-leaf subsplits are compatible if and only if there is at least one leaf in one split that is not in the other.

**Lemma 3.6.** *Let $A|B$ and $C|D$ be compatible four-leaf subsplits. Then $n$, the number of empty intersections*

$$A \cap C,\ A \cap D,\ B \cap C,\ B \cap D$$

*is greater than or equal to $d = |(A \cup B) \ominus (C \cup D)|/2$ (half of the number of different leaves between the subsplits). So,*

$$4 \geq n \geq d.$$

*Proof.* Clearly, the maximum of $d = 4$ which is the only case there $n = 4$ since no leaves are common between $A|B$ and $C|D$. If $d = 1$ then by Lemma 3.5 we know that $n \geq 1 = d$. Thus we only need to consider the cases when $d = 2$ and $d = 3$.

When $d = 2$ then we know $x, y \in A \cup B$ and $x', y' \in C \cup D$ where $x \neq x' \neq y \neq y' \neq x$. Without loss of generality we can prove this lemma for the case $xy = A$ and $x \in A$ and $y \in B$. If $xy = A$, then clearly $A \cap C = A \cap D = \emptyset$ since $x, y \notin C \cup D$. If $x \in A$ and $y \in B$ then we can write $xa = A$. If we let $a \in C$ then we know $a \notin D$ so $A \cap D = \emptyset$. Similarly if $yb = B$ then without loss of generality $b \in C$ so $B \cap D = \emptyset$. Thus $n \geq d$.

Finally when $d = 3$ we can apply the first case of $d = 2$ because $xy = A$ where $x, y \notin C \cup D$ which gives us two empty intersections. Then $zb = B$ where $z \notin C \cup D$ so if $b \in C$ then $B \cap D = \emptyset$ giving us three empty intersections.

Therefore, $4 \geq n \geq d$. □

Even though it is not obvious from the proof that $n$ can be strictly greater than $d$, it is possible. Consider the following example.

*Example* 3.7. Consider 12|34 and 12|35. By Lemma 3.5 we know these subsplits are compatible and $d = 1$. Now to find $n$ we just need to examine the four intersections:

$$12 \cap 12,\ \ 12 \cap 35,\ \ 34 \cap 12,\ \ 34 \cap 35.$$

We have two empty intersections (the second and third) so $n = 2 > 1 = d$.

## 3.2 Constructing a Unique Tree from Four-leaf Subsplits

The first natural question to ask is: which sets of subsplits construct a unique tree? It seems that this question should have an answer since we know compatible splits construct unique trees. In researching this question, I considered the effects of having different partitions in each subsplit.

**Conjecture 3.8.** *A set of four-leaf subsplits will uniquely construct a five-leaf tree if there are at least four different partitions in the subsplits.*

However, this conjecture is false as demonstrated in Example 3.9.

*Counter Example* 3.9. Consider the set of four-leaf splits

$$12|34, \quad 12|35, \quad 12|45.$$

We begin constructing this tree by labeling a four-leaf tree with $1, 2, 3, 4$ in such a way that satisfies the first subsplit. For example, see Figure 3.1. Then using the second and third splits, we want to add the leaf 5 to our tree. Because we know there is an edge between $1, 2$ and $3, 5$ and $4, 5$, there are three possible places we can add the leaf 5: along the branch adjacent to 3, along the branch adjacent to 4, or on the internal edge. Thus, even though we have four different partitions in this set of subsplits, namely 12, 34, 35, and 45, we do not get a unique tree.

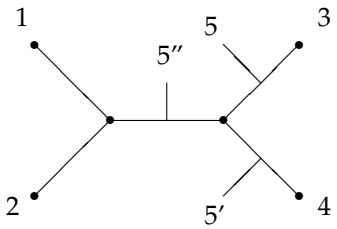



Figure 3.1: Example tree(s) for Counter Example 3.9. First we label the four leaves $1, 2, 3, 4$ then there are three possible locations we can add 5 (where $5, 5', 5''$ are) that would be consistent with the set of given subsplits.

Thus I decided to explore what properties of sets of subsplits would construct a tree. My hope was that this work would lead to insights on the necessary criteria for constructing unique trees from subsplits.

## 3.3 Smallest Set of Unrealizable Four-leaf Subsplits

In the case of splits of $X$, we knew a set of splits would not construct a tree, or were *unrealizable*, if they were not compatible. This is still true with subsplits but we can also have sets of compatible subsplits that are unrealizable.

**Theorem 3.10.** *For $n \geq 5$, the size of the smallest set of compatible four-leaf subsplits that is unrealizable is $n-2$.*

*Proof.* Suppose we want $\{1,2\} \subset X$ to be a cherry. Then we choose another leaf, say 3, and construct a set of $n-3$ subsplits by

$$12|3x$$

where $x \in X \setminus \{1,2,3\}$. This forces a cherry with $\{1,2\}$ because it indicates that there is an edge between $\{1,2\}$ and every other leaf in the tree. Then take two leaves in $X \setminus \{1,2,3\}$, say $4,5$ and add the subsplit $14|25$. Clearly this is compatible with the first $n-3$ subsplits because 3 does not appear in it. However, these subsplits are not realizable because it is trying to place an edge between 1 and 2 which we already said could not exist. Therefore, this set of compatible subsplits is unrealizable. □

## 3.4 $n-3$ Four-leaf Subsplits

Now Theorem 3.10 naturally leads us to question whether every set of $n-3$ subsplits construct a tree. As we discussed in Section 2.2, any set of $n-3$ compatible nontrivial splits of $X$ construct a unique tree. Thus it seems reasonable that this should be the case.

**Conjecture 3.11.** *Every set of $n-3$ compatible four-leaf subsplits constructs a tree.*

My first idea was induction on the number of leaves. We know from graph theory that trees are preserved under addition and subtraction of leaves so the idea was to find a way of adding the appropriate number of subsplits. This lead to a problem of having too many cases to consider. However, from this investigation we learned the following lemma.

**Lemma 3.12.** *Let $x$ be the leaf that appears in the least number of subsplits. If $n$ is the number of times $x$ appears in the set of subsplits, then* $\max n = 3$.

*Proof.* Suppose that the leaf $x$ appears $n = 4$ times. Since there are $4(n-3) = 4n-12$ positions for leaves to occupy in the $n-3$ subsplits and every leaf appears at least 4 times,

$$4n \leq 4n - 12$$

which is a contradiction. □

From this lemma we also see that every leaf can appear at least three times only when $n \geq 12$ and every leaf appears at least twice when $n \geq 6$.

The next idea to prove Conjecture 3.11 was to define a mapping from the set of $n-3$ subsplits to a *subequivalent* set of $n-3$ subsplits where at least one leaf appears in only one subsplit. (By *subequivalent* we mean that the set of trees constructible by the new set of subsplits is a subset of the trees constructible by the original set of subsplits.) If we could do this, then induction would be easy. To accomplish this, we considered choosing a leaf that appeared in two subsplits, left one of the subsplits alone and modified the other so it contains information from both splits. To get a better understanding of this process, see Example 3.13

*Example* 3.13. For the leaf set $X = \{1, 2, 3, 4, 5, 6\}$ consider the subsplits

$$12|34,\ 14|56,\ 25|36.$$

Since 6 appears in the second and third subsplit, let us remove it from the second subsplit. Because 6 appears with 3 in the third split and 3 is not already in the second split, we can just replace 6 in the second split with 3 resulting in the subsplits

$$12|34,\ 14|35,\ 25|36.$$

It is easy to verify that the trees constructed from the latter set of subsplits is a subset of trees constructed from the original set of subsplits.

However, some of the subsplits encode more information so this method does not always work.

*Counter Example* 3.14. Given the same leaf set as in Example 3.13, consider the subsplits

$$12|34,\ 15|46,\ 25|36.$$

This time, let us replace the third subsplits while removing the 6. Following the same procedure as in Example 3.13, we get the subsplits

$$12|34,\ 15|46,\ 25|34.$$

This set of subsplits allows 6 to form a cherry with 1 which was not possible through the original set of subsplits.

Since this method did not work, another idea is to systematically add in the missing leaves to each subsplit. Then we would know what splits we have which in turn could tell us what tree can be constructed. One idea for adding the missing leaves is the following:

Choose a subsplit $A|B$ and choose a leaf $x$ not in that subsplit. In the rest of the subsplits that $x$ appears, compare the relationship of $x$ with every other leaf in $A|B$ as it appears in those subsplits. That is, tally how many times $x$ is opposite elements in $A$ and with elements in $B$ and compare that to the number of times $x$ is opposite elements in $B$ and with elements in $A$. If the first tally wins, add $x$ to $B$ resulting in the (sub)split $A|Bx$, otherwise add it to $A$ resulting in the (sub)split $Ax|B$.

However, this method also does not work completely as revealed by the following counterexample.

*Counter Example* 3.15. Consider the subsplits of $X = \{1,2,3,4,5,6\}$:

$$12|34,\ 15|24,\ 26|13.$$

This set of subsplits happens to uniquely construct the tree in Figure 3.2 with splits $12|3456,\ 1256|34,\ 1234|56$.

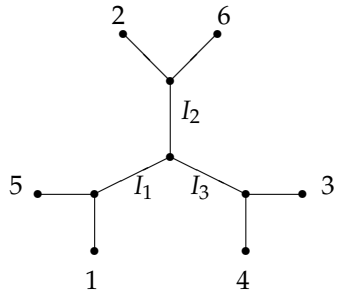



Figure 3.2: Six-leaf tree constructed by the subsplits in Counter Example 3.15

Now we want to add $5,6$ to the first split. Since 5 appears with 1 and opposite $2,4$ in the second subsplit, we want to add 5 to the left side. Similarly with 6, so we have the split $1256|34$. When we add $3,6$ to the second subsplit we end up with the split $135|246$. Clearly this is not consistent with the tree in Figure 3.2 but even worse, it is not compatible with the first split.

Then, when we add $4, 5$ to the last split, we again get $135|246$ so it would seem that this split more likely defines the tree on the other possible six-leaf tree topology which is not the case.

Thus, in order to better understand the problem, I considered adding in the missing leaf to $(n-1)$-leaf subsplits in order to construct splits. As mentioned this would allow me to construct the tree and determine the uniqueness of the tree encoded by the subsplits.

## 3.5 Reconstruction from $(n-1)$-leaf Subsplits

For the rest of this paper, we will only be considering reconstruction from $(n-1)$-leaf subsplits. Because we are no longer dealing with four-leaf subsplits (except when we are considering five-leaf trees), our compatibility rules presented in Section 3.1 and our minimum set of unrealizable subsplits no longer apply. However, the following results are true for $(n-1)$-leaf subsplits.

**Lemma 3.16.** *For $n \geq 6$, every pair of compatible $(n-1)$-leaf subsplits shares at least $n-2$ leaves.*

*Proof.* Let $A|B$ and $C|D$ be compatible $(n-1)$-leaf subsplits. Since they are compatible, we know there exists at least one empty intersection among the sides of both splits. Without loss of generality, say $A \cap C = \emptyset$. Then either $C \subset B$ or the leaf $x$ missing from $A|B$ is in $C$. If $C \subset B$, then $D \subset X \setminus C = A \cup (B \setminus C) \cup \{x\}$. At the most, $D$ will contain one element not in the first subsplit, namely $x$, so the two subsplits will share $n-2$ leaves. Otherwise, if $x \in C$, then the rest of the leaves in $C|D$ must be a subset of the leaves in $A|B$ (because those are the only leaves left to choose from) so $A|B$ and $C|D$ share $n-2$ leaves.

□

We can see how two compatible $(n-1)$-leaf subsplits could actually share all $n-1$ leaves by considering the following. Let $A|B$ be an $(n-1)$-leaf subsplit. Since we are only considering nontrivial subsplits and $n-1 \geq 5$, without loss of generality we can say $|B| > 2$. Let $b \in B$ so $A \cup b|B-b$ has all of the same leaves as $A|B$ and $A \cap (B-b) = \emptyset$ so these subsplits are compatible.

As was mentioned earlier, not every set of compatible subsplits realizes a tree. In fact we can show that there is a minimum size set that will not reconstruct a tree.

**Theorem 3.17.** *The smallest set of compatible $(n-1)$-leaf subsplits that do not realize a tree is three.*

*Proof.* Let $L$ be a set of leaves. Let $a, b \in L$ and $X = L \setminus \{a, b\}$. Then for some $y, z \in X$, the subsplits $ab|X \setminus \{y\}$ and $az|X \setminus \{z\}$ are compatible because $ab \cap X \setminus \{z\} = \emptyset$. These subsplits also force $ab$ to be a cherry since there is an edge between $az$ and $X \setminus \{z\}$ and an edge between $ab$ and $X \setminus \{y\}$ (i.e. $ab|X$). Now introduce the subsplit $ay|b(X \setminus \{y, z\}$ which is compatible with the previous two subsplits since $ay \cap X \setminus \{y\} = \emptyset$ and $az \cap bX \setminus \{y, z\} = \emptyset$. But this subsplit requires us to place an edge between $a$ and $b$ which is not possible since we know $ab$ is a cherry. Therefore this set of three subsplits does not produce an $n$-leaf tree. □

### 3.5.1 Reconstruction Algorithm

Now we will introduce our algorithm for reconstructing trees from subsplit information. The motivation for this process comes from the fact that compatible splits constructs at least one tree. Thus, our goal is to "add" the missing leaf of every $(n-1)$-leaf subsplit in such a way to maintain the compatibility of the splits, if at all possible. (We know there are sets of compatible subsplits that are not realizable as an $n$-leaf tree.) Before we present our algorithm, we need to define a couple of terms.

*Definition* 3.18. A *connection* between two compatible subsplits is a pair $\{S_i^h, S_j^{h'}\}$ where $S_i^h$ and $S_j^{h'}$ are the sides of $S_i$ and $S_j$ that have an empty intersection.

Because every $(n-1)$-leaf subsplit is missing one leaf, we need an idea of which connection is more important or crucial for the compatibility of the subsplits.

*Definition* 3.19. Let $x$ be the missing leaf of subsplit $S_i$. A connection $\{S_i^h, S_j^{h'}\}$ is called *constraining* for $S_i$ if $x \in S_j^{h'}$. We will denote this constraining connection as $\{S_i^h, \overline{S_j^{h'}}\}$.

To get a better understanding of these concepts, consider the compatible subsplits 12|356 and 12|345. The connections between the two subsplits are $\{12, 345\}$ and $\{356, 12\}$. Since 12|356 is missing leaf 4 which is in 345, the first connection is constraining for 12|356. Thus we rewrite the first connection as $\{12, \overline{345}\}$. Similarly, the second connection is constraining for 12|345 so we write that one as $\{\overline{356}, 12\}$.

The following are important lemmas that we will use in proving our main result Theorem 3.23.

**Lemma 3.20.** *Given a set of compatible $(n-1)$-leaf subsplits $\{S_i\}_{i\in\lambda}$, there is at most one connection between $S_i^h$ and $S_j$ for $i, j \in \lambda$ and $h \in \{L, R\}$.*

*Proof.* Let $A|B$ be one of our subsplits. Suppose $A$ had more than one connection with the subsplit $C|D$. That is, $A \cap C = \emptyset$ and $A \cap D = \emptyset$ so the leaves in $A$ cannot be in $C|D$. Notice that $|A| \geq 2$ so there are less than $n-3$ leaves with which to construct $C|D$. This is a contradiction because $C|D$ is an $(n-1)$-leaf subsplit. Hence there is at most one connection between $A$ and the subsplit $C|D$. □

**Lemma 3.21.** *If $n \geq 6$, every pair of compatible $(n-1)$-leaf subsplits does not necessarily have a constraining connection between them.*

*Proof.* If the two subsplits share $n-1$ leaves then any connection cannot be constraining. Otherwise consider the subsplit $A|B$ with $|B| > 2$. Let $x$ be the missing leaf from $A|B$, $a \in A$, and $b \in B$. Then the $(n-1)$-leaf subsplit $(A-a)bx|B-b$ only has a connection between $A$ and $B-b$ which is not constraining. Hence all compatible $(n-1)$-leaf subsplits do not necessarily to have a constraining connection between them. □

Now we are ready to present our $n$-leaf tree reconstruction algorithm.

**Tree Reconstruction Algorithm using $(n-1)$-leaf Subsplits**

Let $\{S_1, \ldots, S_k\}$ be a set of $(n-1)$-leaf compatible subsplits.

1. Let $\mathcal{C}$ be the set of connections between every pair of subsplits with the constraining connections indicated.

2. We will start by adding missing leaves to all subsplits with at least one constraining connection. Suppose $S_i$ has at least one constraining connection. Then we are left with several cases in which we can add the missing leaf to $S_i$:

   (a) If all constraining connections of $S_i$ are associated with the same side of $S_i$ then add the missing leaf to the opposite side. That is, if all constraining connections of $S_i$ are of the form $\{S_i^L, \overline{S_j^{h'}}\}$, then add the missing leaf to $S_i^R$. Repeat Step 2 with the next subsplit.

   (b) If the constraining connections are associated with both sides of $S_i$, then

      i. If $\{S_j\}_{j\in\lambda}$ is the set of all subsplits with a constraining connection associated with $S_i^L$ and there exists another connection $\{S_i^R, S_j^h\}$ for all $j \in \lambda$ then
         - add the missing leaf to $S_i^L$
         - remove all of the constraining connections of the form $\{S_i^L, \overline{S_j^h}\}$
         - add the constraint to any connections as necessary.
         - Repeat Step 2 with the next subsplit.

      ii. If $\{S_j\}_{j\in\lambda}$ is the set of all subsplits with a constraining connection associated with $S_i^R$ and there exists another connection $\{S_i^L, S_j^h\}$ for all $j \in \lambda$ then
         - add the missing leaf to $S_i^L$
         - remove all of the constraining connections of the form $\{S_i^L, \overline{S_j^h}\}$
         - add the constraint to any connections as necessary.
         - Repeat Step 2 with the next subsplit.

      iii. We cannot add in the missing leaf and this set of subsplits is unrealizable as an $n$-leaf tree. End the algorithm.

3. If there are any other subsplits left with no constraining connections, then add the missing leaf to either side.

To ease the proof of Theorem 3.23, we will first prove a helpful lemma that will eliminate some potential problems with adding the missing leaf to a subsplit that is missing the same leaf as another subsplit.

**Lemma 3.22.** *If more than one subsplit is missing leaf $x$, then adding the missing leaf to one of the subsplits will not cause a problem in adding the missing leaf to the others.*

*Proof.* Let $A|B$ and $C|D$ be compatible subsplits both missing leaf $x$. From Lemma 3.16, without loss of generality, $A \subset C$ and $B \supset D$ which means the only connection between the subsplits is $\{A, D\}$. Now consider a third compatible subsplit $X|Y$ such that $x \in X$. Then we need to consider the cases when we have the connection $\{C, \overline{X}\}$ or $\{B, \overline{X}\}$. (If the connections are only associated with $Y$, then there are no constraining connections and adding the missing leaf is trivial.)

Suppose $\{C, \overline{X}\}$ is a constraining connection of $C|D$. Then the algorithm specifies that we add the missing leaf to $D$. This will introduce the constraining connection $\{A, \overline{D}\}$ which could potentially lead to problems. However, notice that since $A \subset C$, there exists the constraining connection $\{A, \overline{X}\}$ so $A|B$ already had a constraining connection with $A|B$, which means we would have still added $x$ to $B$ if $\{A, \overline{D}\}$ were not constraining.

Suppose $\{B, \overline{X}\}$ is a constraining connection of $A|B$. Since $D \subset B$ we know there also exists the constraining connection $\{D, \overline{X}\}$. Now suppose we were in the situation where we had to add $x$ to $D$ (before we add $x$ to $A|B$) so $\{A, \overline{D}\}$ is now constraining. We claim this is still not a problem due to the following. If we were able to add $x$ to $D$, that means there existed a connection $\{C, Y\}$ that allowed us to break the constraining connection $\{D, \overline{X}\}$ without destroying all connections with $X|Y$. Since $A \subset C$, $\{A, Y\}$ also exists so we could add $x$ to $B$ and still maintain a connection between $A|B$ and $X|Y$.

Therefore we can still add the missing leaf to subsplits that are missing the same leaf without concern over the introduction of a constraining connection ending the algorithm unnecessarily. □

Now we are ready to prove our main theorem.

**Theorem 3.23.** *The algorithm terminates after all of the compatible $(n-1)$-leaf subsplits have been transformed into compatible splits if and only if there exists an $n$-leaf tree that displays the subsplits.*

*Proof.* First suppose that the algorithm terminates after all of the subsplits have been transformed into compatible splits. We know that $n-3$ nontrivial compatible splits uniquely define an $n$-leaf tree so any set of compatible

set of splits will define at least one tree. Hence our set of splits will correspond to at least one tree that will display the subsplits.

Now suppose that there exists an $n$-leaf tree that displays the compatible $(n-1)$-leaf subsplits $\{S_i\}_{i\in N}$. We want to show that our algorithm will yield a set of compatible splits that corresponds to this tree. Let $\mathcal{C}$ be the set of connections between the subsplits and let $S_i$ be a subsplit with at least one constraining connection.

If all of the constraining connections are on one side of $S_i$, we are to add the missing leaf to the opposite side. Notice that in this situation we are maintaining all connections (empty intersections) because all constraining connections are still intact and any connections on the opposite side were not constraining so adding the missing leaf to that side did not affect the connections.

If the constraining connections are associated with both sides of $S_i$ then we need to consider all of the subsplits $\{S_j\}_{j\in\lambda}$ that are constraining with one side of $S_i$. Choose $S_i^L$. If $S_i$ has another connection with $S_j$ for all $j \in \lambda$, we know this connection (1) cannot be associated with $S_i^L$ by Theorem 3.20 and (2) cannot be constraining for $S_i$ because the missing leaf cannot appear on both sides of the subsplit. Thus, if we add the missing leaf to $S_i^L$, we have destroyed all of its constraining connections with $S_j$ but still maintain a connection with each $S_j$ through the other one $\{S_i^R, S_j^h\}$ where $h$ is the side of $S_j$ that does not contain the missing leaf of $S_i$.

Notice that we do not need to worry about the introduction of constraining connections because that will only occur if two or more subsplits are missing the same leaf and by Lemma 3.22 we know that no matter what order we add the missing leaf, we will not encounter any unnecessary problems.

If there are any other subsplits left with no constraining connections, then we can add the missing leaf to either side of the subsplit and still maintain all connections.

Notice that since we already know there exists a tree that displays the subsplits, there are associated splits with that tree. Hence if the algorithm were to stop before adding all of the leaves to the subsplits then that would indicate there is not a set of compatible splits that correspond to the tree. Since this is a contradiction, we know that the algorithm will not terminate until all of the subsplits have been transformed into $n$-leaf splits.

Therefore, we have either added the missing leaf into all of the subsplits constructing a set of compatible splits that we know will reconstruction at least one tree or we have shown that the set of subsplits is unrealizable as

an $n$-leaf tree.

□

To get a better understanding of how this algorithm works, consider the following examples.

*Example* 3.24. Consider the 5-leaf subsplits given in Example 3.2:

$$123|45,\ \ 456|23,\ \ 234|16.$$

We will call these splits $S_1, S_2, S_3$, respectfully.

The connections are

$$\{\overline{123}, \overline{456}\},\ \{45, 23\},\ \{\overline{45}, \overline{16}\},\ \{23, \overline{16}\}$$

as illustrated in Figure 3.3.

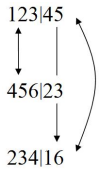



Figure 3.3: The lines in this figure represent the connections between all of the subsplits. The arrows indicate constraining connections for each subsplit with the arrow pointing to the side causing the constraint.

Since each subsplit has a constraining connection, we start by adding 6 to $S_1$. Notice that 6 is connected to both sides of $S_1$ but $S_1$ has two connections with $S_2$. Since we need to maintain the connection with $S_3$, we need to add 6 to the left-side of $S_1$. This will destroy the connection $\{\overline{123}, \overline{456}\}$ but the connection $\{45, 23\}$ is still intact. Now to add 1 to $S_2$ we just need to take into consideration the constraining connection $\{23, \overline{16}\}$ so 1 is forced to be on the left-side. Finally, 5 goes on the left-side of $S_3$ resulting in the splits

$$1236|45,\ \ 1456|23,\ \ 2345|16.$$

Thus this set of subsplits uniquely constructs the 6-leaf tree in Figure 3.4.

Figure 3.3 was a helpful visual aid in determining which subsplits had constraining connections with each other but this type of diagram gets a lot messier with the presence of more subsplits. In these cases, it would be more helpful to the user of the algorithm to use what we shall call a *connection table*.

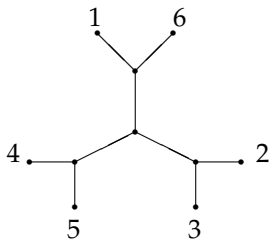



Figure 3.4: The 6-leaf tree that the subsplits in Example 3.24 uniquely constructs.

*Definition* 3.25. Given a set of compatible subsplits $\{S_i\}_{i\in N}$, we can construct a $N \times 3$ *connection table* as follows: Let the $i$th row of the table correspond to $S_i^L$ and $S_i^R$ and the missing leaf of $S_i$, respectfully. In each entry of the table we indicate which subsplit sides the cell of the table has a connection. We indicate the constraining connections with a bar over the subsplit side.

The following example will use this notion of a connection table to aid in the implementation of the algorithm.

*Example* 3.26. Let the following be our collection of subsplits with which we want to construct an eight-leaf tree:

$$128|3456,\ 357|2468,\ 37|12468,\ 57|12368,\ 126|3457.$$

We will refer to these subsplits as $A, B, C, D, E$, respectfully. As the algorithm specifies, we first need to consider all of the connections. To help us with this task, we will use a connection table.

| | $L$ | $R$ | Missing leaf |
|---|---|---|---|
| $A$ | $\overline{B^L}\ \overline{C^L}\ \overline{D^L}\ E^R$ | | 7 |
| $B$ | $\overline{A^L}\ \overline{C^R}\ \overline{E^L}$ | $C^L\ D^L$ | 1 |
| $C$ | $A^L\ B^R\ E^L$ | $\overline{B^L}\ \overline{D^L}$ | 5 |
| $D$ | $A^L\ \overline{B^R}\ \overline{C^R}\ E^L$ | | 4 |
| $E$ | $B^L\ C^L\ D^L$ | $\overline{A^L}$ | 8 |

Since all of these subsplits have at least one constraining connection, we will add the missing leaves in order (from $A$ to $E$). Notice that in the first row of our connection table all of the constraining connections are on the

left side of $A$ so we will add the missing leaf to the right side. Similarly for the rest of the subsplits, all of the constraining connections are on one side of the subsplit so we can add the missing leaf to the opposite side. This leaves us with the splits

$$128|34567,\ 357|12468,\ 357|12468,\ 57|123468,\ 1268|3457.$$

These splits correspond to the tree in Figure 3.5. Notice that on the far right of the tree there is some ambiguity in the relationship between leaves 1, 2, and 8. This is because the second and third subsplits were transformed into the same split so we could not uniquely construct a tree from this set of subsplits.

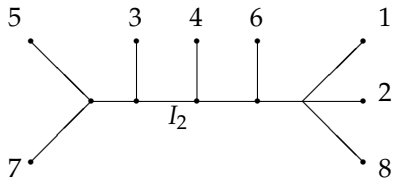



Figure 3.5: This is the eight-leaf tree that displays the subsplits in Example 3.26. Notice that the right side of the tree where 1, 2, and 8 branch is still ambiguous because the second and third subsplits correspond to the same edge $I_2$.

Now that we have seen how this algorithm indicates when a set of compatible $(n-1)$-leaf subsplits will reconstruct a tree, it would be beneficial to look at a set of compatible subsplits that are unrealizable as an $n$-leaf tree.

*Counter Example* 3.27. Now consider the splits in Counter Example 3.3.

The connections for these subsplits are

$$\{\overline{345}, \overline{16}\},\ \{\overline{12}, \overline{346}\},\ \{\overline{234}, \overline{15}\}$$

each of which is constraining for both subsplits involved. Let us start by trying to add the leaf 6 to the first subsplit. Notice that the left-side 12 is connected to 6 with the third subsplit (in the second connection) and the right-side 345 is connected to a 6 with the second subsplit (in the first connection). Since the first subsplit has no other connections with either subsplit, then there is no way we can add 6 and maintain compatibility between all of the subsplits. Thus, this set of compatible subsplits is unrealizable as a six-leaf tree.

Without using the algorithm, we can also see how this set of subsplits is not displayed by a tree. The first subsplit 12|345 indicates that there exists an edge between 12 and 345. Similarly the second subsplit 16|234 indicates an edge between 16 and 234 so these conditions force 16 to be a cherry of any tree that displays these subsplits. However, the last subsplit 15|346 is trying to place an edge between 1 and 6 which we know to be impossible if 16 is a cherry. Hence this set of subsplits do not realize a six-leaf tree.

As we mentioned in the literature review, Semple and Steel's conditions for uniquely constructing a tree from partial partitions requires that we know if the given set of partial partitions is consistent with any tree. (22) The algorithm presented in this section precisely answers this question in the case that the partial partitions are subsplits of size $n-1$. Thus this work can be thought of as a tool to use Semple and Steel's existence of a unique tree theorem. From the algorithm we can tell if the tree is unique because our use of constraining connections is precisely the chordation idea presented in Semple and Steel's work (the constraining connections can be thought of as a modified inverse of the minimum restricted chordal completion of Semple and Steel's partial partition intersection graph). So not only is our work helpful for applying Semple and Steel's work allowing us to construct the tree and check for uniqueness, but also tells us if there are multiple trees that satisfy the subsplit constraints where as Semple and Steel only consider the existence of a unique tree.

# Chapter 4

# Conclusion

All of the research presented in this paper has been conducted in the hopes of aiding the reconstruction of phylogenetic trees and to gain a better understanding of the mathematics needed to build such results. We first considered reconstruction from four-leaf subsplits and proved some helpful facts about these subsplits. From there we adjusted our focus to reconstruction from $(n-1)$-leaf subsplits. The algorithm presented in Section 3.5.1 allows us to determine when a set of subsplits can reconstruct a unique tree or set of trees. (Recall that a set of $n-3$ nontrivial compatible splits will uniquely reconstruct an $n$-leaf tree.) We can use this algorithm to develop a constructive proof of other subsplit results such as the minimum number of compatible $(n-1)$-leaf subsplits that is unrealizable as an $n$-leaf tree. This algorithm also can be used as a construction method for the first criteria necessary in Semple and Steel's unique reconstruction of trees from partial partitions of the leaves.

Further questions to explore include: can this method be generalized to $(n-m)$-leaf subsplits? In the case that the topology of the tree is unique, can we use this method to construct unique weighted trees? Is this work applicable to reconstruction of phylogenetic trees, i.e., is it easier for biologist to collect subsplit data rather than split data?

# Appendix A

# Paper: “The Shapley Value of Phylogenetic Trees”

The Shapley Value of Phylogenetic Trees

Draft 2/23/2005

Claus-Jochen Haake]Claus-Jochen Haake: Institute of Mathematical Economics, Bielefeld University, PO Box 10 01 31, 33501 Bielefeld, Germany, chaake@wiwi.uni-bielefeld.de

Akemi Kashiwada: Research partially supported by a Howard Hughes Medical Institute Undergraduate Science Education Program grant to Harvey Mudd College.

Francis Su:Research partially supported by NSF Grant DMS-0301129.

Department of Mathematics, Harvey Mudd College, Claremont, CA 91711, U.S.A., akashiwada@hmc.edu, su@math.hmc.edu

Abstract: Every weighted tree corresponds naturally to a cooperative game that we call a *tree game*; it assigns to each subset of leaves the sum of the weights of the minimal subtree spanned by those leaves. In the context of phylogenetic trees, the leaves are species and this assignment captures the *diversity* present in the coalition of species considered. We consider the Shapley value of tree games and suggest a biological intepretation. We determine the linear transformation $\mathbf{M}$ that shows the dependence of the Shapley value on the edge weights of the tree, and we also compute a null space basis of $\mathbf{M}$. Finally, we characterize the Shapley value on tree games by five axioms, a counterpart to Shapley's original theorem on the larger class of cooperative games. We also include a brief discussion of the core of tree games.

## A.1 Introduction

The *Shapley value* is arguably the most important solution concept for $n$-player cooperative games. Given a set of players $N$ in a cooperative game $v$, the Shapley value $\varphi(N, v)$ is the unique imputation vector that satisfies four "fairness" criteria (the *Shapley axioms*) that we shall discuss later. In this paper we consider the game $v_{\mathcal{T}}$ induced by an unrooted $n$-leaf tree $\mathcal{T}$ in which each edge is assigned a positive number called an *edge weight*. In this context, the players are represented by the leaves of the tree and the value of any coalition $S$ is the total weight of the subtree spanned by the members of $S$.

In a more applied context, we consider games induced by a *phylogenetic tree* in which players are species and the tree represents a proposed evolutionary relationship among the species. We suggest that a biological

interpretation for the Shapley value is a notion of the average marginal diversity that a species brings to any group, and we study how the Shapley value depends on the edge weights and topology of the tree.

One possible application of the Shapley value of a phylogenetic tree is the economic theory of biodiversity preservation (20; 25). The *Noah's ark problem* (26) asks how to prioritize species in a population if only some limited number can be saved; we suggest that Shapley value provides a natural ranking criterion.

The literature applying game-theoretic solution concepts to an analysis of trees appears to be limited. One closely related example is Kar (17), who studies cost-sharing in a network structure and characterizes the Shapley value of the minimum cost spanning tree game of an arbitrary graph. However, his work differs from ours because he considers each node of a graph as a player in the game, whereas we specifically study tree games and allow only leaves as players. Day and McMorris (11) propose suitable axioms for a consensus rule that will aggregate several phylogenetic trees into one consensus tree; this differs from the thrust of our work, which is to consider one tree and explore the interpretation and properties of the Shapley value of the associated tree game.

In the next section we provide a biological interpretation for the Shapley value of phylogenetic trees. Then we discuss the mathematics of calculating the Shapley value on tree games, starting with some examples on small trees. In the subsequent section we present several theorems demonstrating how the Shapley value of an $n$-player game can be calculated from its $n - m$ player subgames. We also examine the null space of the Shapley value with respect to the tree topology. In Section A.6, we take a brief look at the core of tree games. We conclude this paper by developing an analogue of Shapley's theorem that characterizes the Shapley value on games by four axioms. We show that on the smaller class of tree games, the Shapley value is characterized by those four axioms plus an additional axiom.

## A.2 Phylogenetic Trees and the Shapley Value

### A.2.1 Phylogenetic trees

Evolutionary relationships between species are frequently represented by a *phylogenetic tree*. Evidence for such relationships can come from a variety of sources, such as genomic data or morphological comparisons, and much work has been done to develop methods for constructing a phylogenetic

tree from such data (for surveys, see Felsenstein (14) and Semple-Steel (23)).

Phylogenetic trees are usually binary trees in which each internal node represents a bifurcation in some characteristic and the leaves are the species for which we have data. Each edge has a weight that represents some unit of distance between the nodes at its endpoints (for instance, it could be the time between speciation events). Figure A.1 gives a small example of what a (rooted) phylogenetic tree could look like. However, in this paper we shall not be concerned with the location of the root of a tree, so all our trees will be unrooted.

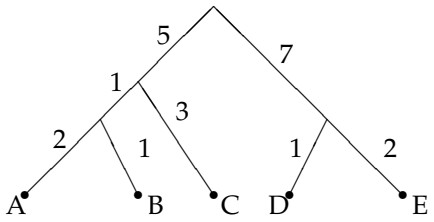



Figure A.1: Example of a phylogenetic tree with species A-E with edge weights labeled.

Formally, we shall think of a phylogenetic tree $\mathcal{T}$ as an unrooted tree with leaf set $N := \{1, \ldots, n\}$ (representing the species in the population), edge set $E$, and and an edge weight $\alpha_k$ for each edge $k$ in $E$.

### A.2.2 The Shapley value

In cooperative game theory, a *cooperative game* is a pair $(N, v)$ consisting of a set of *players* $N = \{1, 2, ..., n\}$ and a *characteristic function* $v$ that takes every subset of $N$ (called a *coalition*) to a real number (called the *worth* of the coalition). The subset consisting of all players is called the *grand coalition*. Formally, if $2^N$ is the set of all subsets of $N$, then $v : 2^N \to \mathbb{R}$.

For instance, $N$ could be a set of companies and $v$ could describe the profit that each coalition of companies could make if the members of that coalition worked together. Usually, game theorists are interested in *superadditive* games in which $v(S \cup T) \geq V(S) + V(T)$ for any two coalitions $S$ and $T$. In such games, there is incentive for players to cooperate when the inequality is strict. One of the basic questions in cooperative game theory is: if players work together to achieve some total worth (in our exam-

ple, profit), how should players then distribute their worth (profit) among themselves?

As all (Pareto efficient) solution concepts from cooperative game theory do, the *value* introduced by Shapley (24) suggests a "fair" distribution of the total worth of the entire set of players $N$ among the members of $N$. Given a cooperative game $(N, v)$, the Shapley value is a vector $\varphi = (\varphi_i)$ defined by the formula

$$\varphi_i(N, v) = \frac{1}{n!} \sum_{\substack{S \subseteq N \\ i \in S}} (s-1)!(n-s)!(v(S) - v(S-i)) \tag{A.1}$$

where $s = |S|$ is the size of the coalition $S$ and $n = |N|$ is the total number of players.

The formula above has a sensible interpretation that suggests a rationale for the Shapley value to obtain a "fair" distribution. For a player $i \in N$ and a coalition $S \subseteq N$ that contains $i$, the quantity $v(S) - v(S-i)$ describes $i$'s marginal contribution to the worth of $S$. If we choose a random ordering of the players, and consider the growing coalition that forms when the players are added one-by-one from that ordering, then the combinatorial form of (A.1) reflects the Shapley value's interpretation as an *average of the marginal contributions* that $i$ makes to a randomly chosen coalition.

### A.2.3 The Phylogenetic Tree Game

Given a phylogenetic tree $\mathcal{T}$, we can define an associated cooperative game $(N, v_{\mathcal{T}})$ that we call a *phylogenetic tree game*. Let $N$ be the set of leaves of the tree (species). For any subset $S \subseteq N$ of species, consider the unique spanning subtree containing the members in $S$, and let $v_{\mathcal{T}}(S)$ be the sum of the edge weights of that spanning tree. Thus for each set $S$ we may think of $v_{\mathcal{T}}(S)$ a measure of the *diversity* within $S$. This forms a cooperative game $(N, v_{\mathcal{T}})$ in a natural way, and it is evident from our definition that the phylogenetic tree game is superadditive.

Although species can hardly be compared with rationally acting agents (as usually assumed in theory of cooperative games), we may still ask for a meaningful re-interpretation of game-theoretic solution concepts such as the Shapley value in the context of phylogenetic trees.

Given a phylogenetic tree game $(N, v_{\mathcal{T}})$, equation (A.1) suggests that the Shapley value of a given species may be thought of as its *average marginal diversity*, i.e., the average diversity the species can be expected to add to a

group that it joins. So if $\varphi_i > \varphi_j$, then species $i$ can be thought to contribute a greater diversity to a group than species $j$ might.

### A.2.4 The Shapley Value Axioms

Besides the interpretation of the Shapley value as an average expected marginal contribution, there is an axiomatization of the Shapley value (see (24)) that uniquely characterizes it by a set of (desirable) properties. We review the axioms presented by Shapley and discuss their plausibility in the present setting as properties of phylogenetic trees. Let therefore $\mathcal{V} := \left\{v : 2^N \to \mathbb{R} \mid v(\emptyset) = 0\right\}$ be the set of all cooperative games with $n$ players.

1. (*Pareto Efficiency Axiom*) The Shapley value is Pareto efficient, i.e., $\sum_{i \in N} \varphi_i(N, v) = v(N)$ for all $v \in \mathcal{V}$.

   This axiom just states that the total diversity present within a phylogenetic tree will be distributed and ascribed to the species within it. This is a reasonable axiom, given that the purpose of a solution concept for a cooperative game is to distribute the worth of the grand coalition among its members. In this context, the natural interpretation is that the Shapley value answers the question of how much a specific species is responsible for the total diversity, or, put another way, what is its *share* of $v_{\mathcal{T}}(N)$.

2. (*Symmetry Axiom*) For any permutation of players $\pi : N \to N$ the Shapley value satisfies $\varphi(\pi v) = \pi\varphi(v)$, where $\pi v$ is the permuted game given by $\pi v(S) := v(\pi^{-1}(S))$ for all $S \subseteq N$ and $\pi\varphi(v)$ is the permuted solution vector, i.e., $(\pi\varphi(v))_i := \varphi_{\pi^{-1}(i)}(v)$.

   The symmetry axiom states that a player's allocation should not be based on her name. Another consequence of the symmetry axiom is if exchanging two players causes no difference in the worth that each adds to any coalition, then they should have the same Shapley value. Biologically speaking, if two species play the same role within a tree then they should be ascribed the same responsibility for diversity, which seems to be a plausible requirement.

3. (*Dummy Axiom*) A dummy player is one that does not add worth to the value of any coalition. This axiom says that dummy players should have a Shapley value of zero.

   This axiom is vacuously satisfied in the case of a phylogenetic tree game because there are no dummy species. To see this, note that

every species $i$ adds worth to the coalition that consists of a single species $j \neq i$, because the weight of the subtree containing $i$ and $j$ is the sum of the edge weights between $i$ and $j$ and is therefore non-zero, but the weight of the subtree consisting of the singleton $j$ is zero. (Even though there are no dummy species, this is still a reasonable axiom here, since any species that does not diversify any coalition should get value zero.)[1]

4. (*Additivity Axiom*) Given two games $(N, v)$ and $(N, w)$ in $\mathcal{V}$ with the same set of players $N$, define the *sum game* $(N, v + w)$ with characteristic function $(v + w)(S) = v(S) + w(S)$ for every coalition $S$. This axiom stipulates that the Shapley value of the sum game should be the sum of the Shapley values of the individual games: $\varphi(N, v + w) = \varphi(N, v) + \varphi(N, w)$.

   As an example, suppose we are given genome sequences for a set of species $N$, and each sequence has length 200. For each pair of species $i, j$ consider the (rather crude) measure of distance $d(i, j)$ to be the number of positions in which the sequences differ. The pairwise distance data can be used to construct a tree (using any standard method) and consequently, a tree game. Thus the first 100 positions of the sequences can be used to construct a tree game $(N, v_1)$, and the second 100 positions a tree game $(N, v_2)$. Then the Shapley value of the sum game $(N, v_1 + v_2)$ is the sum of the Shapley values for each game. This seems plausible in this context, since if the pairwise distances $d(i, j)$ from both sets of 100 positions actually arise from a tree metrics on the same tree, then the sum game will arise from the tree reconstructed from all 200 positions.

## A.3 Examples and Motivation: the Shapley Value for Small Trees

As can be seen from (A.1), the Shapley value of a tree game is a linear function of the edge weights of the tree. We call that linear transformation the *Shapley transformation*. Before deriving a general formula for this transformation in the subsequent section, we study the Shapley transformation for games induced by unrooted three-, four-, five- and six-leaf trees.

[1] In Section A.5 we will replace the dummy axiom by a different one to characterize the Shapley value on the class of games that actually come from trees.

We will refer to the weights of edges incident to leaves as *leaf weights* and other edge weights as *internal edge weights*. Note that for an unrooted $n$-leaf tree, there are $n-2$ internal nodes and $n-3$ internal edges in $E$. In what follows, the superscript $^T$ denotes the *transpose*.

*Definition* A.1. Let $\mathcal{T}$ be an $n$-leaf tree with leaves $N = \{1, \ldots, n\}$, associated leaf weights $\alpha_1, \ldots, \alpha_n$ and internal edges $I_1, \ldots, I_{n-3}$ with associated internal edge weights $\alpha_{I_1}, \ldots, \alpha_{I_{n-3}}$. Let $\vec{E}$ be a vector consisting of the edge weights in this order: $(\alpha_1, ..., \alpha_n, \alpha_{I_1}, ..., \alpha_{I_{n-3}})^T$. Define $\mathbf{M} = \mathbf{M}(N, v_T)$ to be the $n \times (2n-3)$ matrix that represents the Shapley transformation, so that the Shapley value of the game $v_\mathcal{T}$ is

$$\varphi(N, v_\mathcal{T}) = (\varphi_1, \varphi_2, \ldots, \varphi_n)^T = \mathbf{M}\vec{E}$$

where $\varphi_i$ is the Shapley value associated with leaf $i$. Note that $\mathbf{M}$ depends on the topology of the $n$-leaf tree.

Later we will determine a formula for $\mathbf{M}[i,k]$, which is the coefficient of edge weight $k$ in the calculation of the Shapley value of $i$. But first, we give a few examples.

### A.3.1 Three-Leaf Trees

Topologically, there is only one unrooted three-leaf tree $\mathcal{T}$. Let the leaves represent players A, B, and C with corresponding leaf weights $\alpha$, $\beta$, and $\gamma$ as seen in Figure A.2.

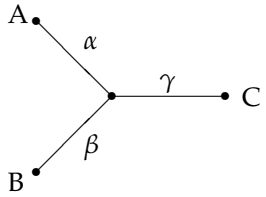



Figure A.2: The topology of an unrooted three-leaf tree $\mathcal{T}$ where the players are A, B, and C with corresponding leaf weights $\alpha$, $\beta$, and $\gamma$.

The characteristic function $v_\mathcal{T}$ for this game is

$$v_\mathcal{T}(A) = v_\mathcal{T}(B) = v_\mathcal{T}(C) = 0,$$

$$v_\mathcal{T}(AB) = \alpha + \beta, \;\; v_\mathcal{T}(AC) = \alpha + \gamma, \;\; v_\mathcal{T}(BC) = \beta + \gamma,$$

$$v_T(ABC) = \alpha + \beta + \gamma.$$

Using Definition A.1, we can calculate the Shapley value by $\varphi = (\varphi_A, \varphi_B, \varphi_C) = \mathbf{M}\vec{\ell}$ where $\vec{\ell}$ is the vector of leaf weights $(\alpha, \beta, \gamma)^T$ and

$$\mathbf{M} = \frac{1}{6}\begin{bmatrix} 4 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 4 \end{bmatrix}.$$

It is apparent that we can solve for $\alpha$, $\beta$, and $\gamma$ in terms of $\varphi$ by inverting $\mathbf{M}$:

$$\vec{\ell} = \frac{1}{3}\begin{bmatrix} 5 & -1 & -1 \\ -1 & 5 & -1 \\ -1 & -1 & 5 \end{bmatrix}\begin{pmatrix} \varphi_A \\ \varphi_B \\ \varphi_C \end{pmatrix}.$$

This means the Shapley value of a 3-leaf tree uniquely determines the tree representing the game.

### A.3.2 Four- and Five-Leaf Trees

Using the same procedure as in the three-leaf tree case, we can calculate the Shapley value for each player in the four- and five-leaf case. There is a unique tree topology for each case as shown in figure A.3.

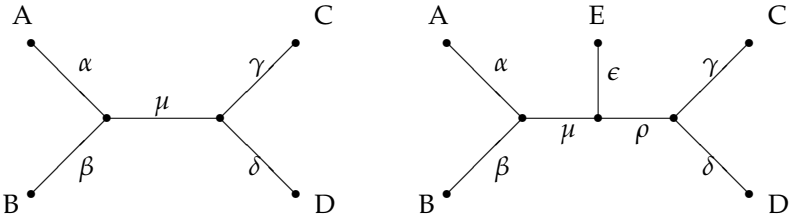



Figure A.3: (*left*)The topology for an unrooted four-leaf tree where the players are A, B, C, and D. (*right*) The unrooted five-leaf tree with players A, B, C, D, and E.

The Shapley value for the general four-leaf tree game is

$$\frac{1}{24}\begin{bmatrix} 18 & 2 & 2 & 2 & 6 \\ 2 & 18 & 2 & 2 & 6 \\ 2 & 2 & 18 & 2 & 6 \\ 2 & 2 & 2 & 18 & 6 \end{bmatrix}\begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \mu \end{pmatrix}.$$

Similarly for the five-leaf tree game, the Shapley value is

$$\frac{1}{120}\begin{bmatrix} 96 & 6 & 6 & 6 & 6 & 36 & 16 \\ 6 & 96 & 6 & 6 & 6 & 36 & 16 \\ 6 & 6 & 96 & 6 & 6 & 16 & 36 \\ 6 & 6 & 6 & 96 & 6 & 16 & 36 \\ 6 & 6 & 6 & 6 & 96 & 16 & 16 \end{bmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \epsilon \\ \mu \\ \rho \end{pmatrix}.$$

It is apparent from the fact that there are more variables (edge weights) than equations that there is not a unique set of (possibly negative) edge weights for a given Shapley value. That is, there is not a unique tree corresponding to a given Shapley value. The null space of **M** will therefore help us determine which weighted trees have the same Shapley value. A basis for the null space of **M** for the four-leaf tree is

$$\left\{ \begin{pmatrix} -1/4 \\ -1/4 \\ -1/4 \\ -1/4 \\ 1 \end{pmatrix} \right\}$$

This means that given a tree $\mathcal{T}$, we can produce other trees with the same Shapley value by reducing the leaf weights by $1/4$ for each unit increase in the internal edge weight.

Similarly, a null space basis for the five-leaf tree is

$$\left\{ \begin{pmatrix} -1/3 \\ -1/3 \\ -1/9 \\ -1/9 \\ -1/9 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1/9 \\ -1/9 \\ -1/3 \\ -1/3 \\ -1/9 \\ 0 \\ 1 \end{pmatrix} \right\}.$$

### A.3.3 Six-Leaf Trees

For our last direct calculation, let us consider the games represented by six-leaf trees. In this case there are two topologies for unrooted trees with six leaves (see figure A.4).

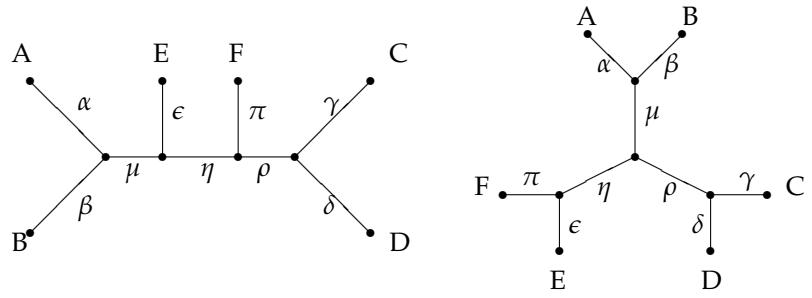



Figure A.4: (*left*)The first topology for an unrooted six-leaf tree $\mathcal{T}$ where the players are A, B, C, D, E and F. (*right*) The second unrooted six-leaf tree $\mathcal{T}'$.

The Shapley value for the first and second six-leaf trees are, respectively,

$$\varphi(N, v_{\mathcal{T}}) = \frac{1}{720}\begin{bmatrix} 600 & 24 & 24 & 24 & 24 & 24 & 240 & 60 & 120 \\ 24 & 600 & 24 & 24 & 24 & 24 & 240 & 60 & 120 \\ 24 & 24 & 600 & 24 & 24 & 24 & 60 & 240 & 120 \\ 24 & 24 & 24 & 600 & 24 & 24 & 60 & 240 & 120 \\ 24 & 24 & 24 & 24 & 600 & 24 & 60 & 60 & 120 \\ 24 & 24 & 24 & 24 & 24 & 600 & 60 & 60 & 120 \end{bmatrix}\begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \epsilon \\ \pi \\ \mu \\ \rho \\ \eta \end{pmatrix},$$

$$\varphi(N, v_{\mathcal{T}'}) = \frac{1}{720}\begin{bmatrix} 600 & 24 & 24 & 24 & 24 & 24 & 240 & 60 & 60 \\ 24 & 600 & 24 & 24 & 24 & 24 & 240 & 60 & 60 \\ 24 & 24 & 600 & 24 & 24 & 24 & 60 & 240 & 60 \\ 24 & 24 & 24 & 600 & 24 & 24 & 60 & 240 & 60 \\ 24 & 24 & 24 & 24 & 600 & 24 & 60 & 60 & 240 \\ 24 & 24 & 24 & 24 & 24 & 600 & 60 & 60 & 240 \end{bmatrix}\begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \\ \epsilon \\ \pi \\ \mu \\ \rho \\ \eta \end{pmatrix}.$$

As with the four and five leaf cases, both topologies of the six leaf tree allow for many trees to possess the same Shapley value. The basis for the

null space of the first six-leaf tree is

$$\left\{ \begin{pmatrix} -3/8 \\ -3/8 \\ -1/16 \\ -1/16 \\ -1/16 \\ -1/16 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -1/16 \\ -1/16 \\ -3/8 \\ -3/8 \\ -1/16 \\ -1/16 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1/6 \\ -1/6 \\ -1/6 \\ -1/6 \\ -1/6 \\ -1/6 \\ 0 \\ 0 \\ 1 \end{pmatrix} \right\}$$

and for the second six-leaf tree is

$$\left\{ \begin{pmatrix} -3/8 \\ -3/8 \\ -1/16 \\ -1/16 \\ -1/16 \\ -1/16 \\ 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} -1/16 \\ -1/16 \\ -3/8 \\ -3/8 \\ -1/16 \\ -1/16 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -1/16 \\ -1/16 \\ -1/16 \\ -1/16 \\ -3/8 \\ -3/8 \\ 0 \\ 0 \\ 1 \end{pmatrix} \right\}.$$

### A.3.4 Notes on Relationship between Trees and Shapley Values

¿From these examples, we make a few observations.

1. Any Shapley value $n$-vector can be realized by adjusting the edge weights of an $n$-leaf tree. This may involve positive as well as nonpositive edge weights. However, the positive hull of the column vectors of the matrix $\mathbf{M}$ can be realized as the Shapley value of some tree with nonnegative edge weights.

2. When $n \geq 4$, there is not a unique $n$-leaf tree corresponding to a given Shapley value because the null space is nontrivial.

3. The null space bases for the two six-leaf trees are different; hence enough to determine the topology of the tree. As we shall see in the next section, this phenomenon is true in general.

4. Under close inspection, one notices a relationship between the numbers of leaves on each side of an internal edge and quantities such

as the entries of the Shapley transformation matrix and the null space basis vectors. We exhibit their explicit dependence in the next section.

## A.4 Calculating the Shapley Value from Subtrees

In this section, we shall prove that the Shapley value for an $n$-leaf tree game can be calculated from the Shapley value of all its $(n-m)$-leaf subtree games. First we will show that given a tree game with $n$ players, the Shapley values can be calculated from the Shapley value for all $(n-1)$-leaf subtrees. This kind of "reconstruction" result stands in stark contrast to a result of Pachter-Speyer (Pachter and Speyer) for trees; they show that an $n$-leaf tree cannot necessarily be reconstructed from the weights of its $(n-1)$-leaf subtrees.

We first show the contribution of each edge weight to the Shapley value; these are the entries of the matrix representing the Shapley transformation.

### A.4.1 Entries in Shapley Value Matrix

The following theorem gives us a quick way of finding the $(i,k)$th entry of the Shapley value matrix of an $n$-leaf tree game. Before we state and prove the theorem, we need to present a definition that is instumental throughout the rest of this paper.

*Definition* A.2. Let $\mathcal{T}$ be an $n$-leaf tree with leaves $N$ and edges $E$. For $i \in N$ and $k \in E$, the removal of edge $k$ splits $\mathcal{T}$ into two subtrees. Let $\mathcal{C}(i,k)$ denote the subtree that *contains* $i$ (the "containing" subtree) and let $\mathcal{F}(i,k)$ denote the subtree that is "far" from $i$. We then denote the number of leaves of $\mathcal{C}(i,k)$ and $\mathcal{F}(i,k)$ as $c(i,k)$ and $f(i,k)$, respectfully.

If it is obvious what leaf $i$ and edge $k$ we are referring to, we will simply write $c, f$ instead of $c(i,k), f(i,k)$. Note that $n = c + f$. We call $c, f$ the *split counts* associated with leaf $i$ and edge $k$. As we shall see, the split counts will arise frequently in our results on the Shapley transformation.

**Theorem A.3.** *Let $\mathcal{T}$ be an $n$-leaf tree. The $(i,k)$th entry of the Shapley transformation matrix* $\mathbf{M}$ *is given by*

$$\mathbf{M}[i,k] = \frac{f(i,k)}{n\, c(i,k)}.$$

*Proof.* It is sufficient to show this theorem is true in calculating the Shapley value of a single leaf in the $n$-leaf tree game. Fix leaf $i$. To count the number

of times a given edge weight contributes to $i$'s Shapley value, we need to know how many times it is in the marginal contribution of $i$ for coalitions of size $s$. Edge weight $\alpha_k$ will be part of $i$'s marginal contribution if the other $s-1$ members of the coalition are from the opposite side of the edge from $i$. So

$$\mathbf{M}[i,k] = \frac{1}{n!}\sum_{s=2}^{n}(s-1)!(n-s)!\binom{f(i,k)}{s-1} = \frac{1}{n!}\sum_{s=2}^{n}\frac{(n-s)!f(i,k)!}{(f(i,k)-s+1)!}.$$

Using the fact $f = n - c$, the above expression can be rewritten:

$$\frac{1}{n!}\sum_{s=2}^{n}(n-c)!(c-1)!\binom{n-s}{c-1} = \frac{(n-c)!(c-1)!}{n!}\sum_{j=1}^{n-1}\binom{j-1}{c-1}.$$

We use the identity

$$\sum_{j=1}^{n}\binom{j-1}{c-1} = \binom{n}{c} = \binom{n-1}{c-1}\frac{f}{c} + \binom{n-1}{c-1}$$

to obtain

$$\mathbf{M}[i,k] = \frac{(n-c)!(c-1)!}{n!}\binom{n-1}{c-1}\frac{f}{c} = \frac{f}{nc}.$$

□

This result is particularly nice because it shows how the Shapley value's dependence on any edge weight only depends on the number of leaves on either side of that edge. Consider the following example.

*Example* A.4. Using Theorem A.3 we will calculate the coefficient of $\mu$ in player A's Shapley value for a five-leaf tree. Let the edge with edge weight $\mu$ be $I_1$. There are three leaves in $\mathcal{F}(A, I_1)$ and two leaves in $\mathcal{C}(A, I_1)$. Thus,

$$\mathbf{M}[1,6] = \frac{3}{5\cdot 2}$$

which is the same as the $(A,\mu)$ entry $36/120$ in the Shapley transformation of the five-leaf tree given in section A.3.2.

With the above result we can calculate the Shapley value of an $n$-leaf tree game from the Shapley value of subtree games.

### A.4.2 Shapley Value from $(n-1)$-leaf Subtrees

In this section we want to show how the Shapley value of an $n$-leaf tree game can be calculated from the Shapley value of its $(n-1)$-leaf subtrees. Before we can do that we need the following definition and lemma.

*Definition* A.5. Let $\mathcal{T}$ be an $n$-leaf tree with leaves $N$. For any subset of leaves $S \subseteq N$, the Shapley value of $N$ with respect to the subtree spanned by $S$ is $\varphi(N, v_{S,\mathcal{T}}) \in \mathbb{R}^n$ where for any coalition $U \subseteq N$, $v_{S,\mathcal{T}}(U) = v_{\mathcal{T}}(S \cap U)$. Put another way, $\varphi(N, v_{S,\mathcal{T}}) = \varphi(S, v_{S,\mathcal{T}}) \times \{0\}^{N \setminus S}$.

In other words, this Shapley value will assign zero to any player not in $S$ and the usual Shapley value of the tree game spanned by $S$ to players in $S$.

**Lemma A.6.** *Let $\mathcal{T}$ be an $n$-leaf tree with leaves $N = \{1, \ldots, n\}$ and internal edges $I_1, \ldots, I_{n-3}$ with corresponding edge weights $\alpha_{I_1}, \ldots, \alpha_{I_{n-3}}$. Fix $i \in N$. Then*

$$\sum_{\substack{i \in S \subset N \\ |S| = n-1}} \sum_{k=1}^{n-3} \frac{f_S(i,k)}{c_S(i,k)} \alpha_{I_k} = (n-1) \sum_{k=1}^{n-3} \frac{f_N(i,k)}{c_N(i,k)} \alpha_{I_k}$$

*where $f_U, c_U$ are determined with respect to leaf $i$ in the tree spanned by $U \subseteq N$.*

*Proof.* Fix $k \in \{1, \ldots, n-3\}$. Notice that each subset $S$ is obtained by deleting one leaf $j \neq i$ either from $\mathcal{F}(i, I_k)$ or from $\mathcal{C}(i, I_k)$. Since $|\mathcal{F}(i, I_k)| = f_N$ and $|\mathcal{C}(i, I_k)| = c_N - 1$,

$$\sum_{\substack{i \in S \subset N \\ |S| = n-1}} \frac{f_S}{c_S} \alpha_{I_k} = \left( f_N \frac{f_N - 1}{c_N} + (c_N - 1) \frac{f_N}{c_N - 1} \right) \alpha_{I_k} = (n-1) \frac{f_N}{c_N} \alpha_{I_k}.$$

Thus, summing over all $k$, we obtain

$$\sum_{\substack{i \in S \subset N \\ |S| = n-1}} \sum_{k=1}^{n-3} \frac{f_S}{c_S} \alpha_{I_k} = (n-1) \sum_{k=1}^{n-3} \frac{f_N}{c_N} \alpha_{I_k}.$$

□

Now we are ready to show how we can calculate the Shapley value of an $n$-leaf tree from the Shapley values for all its $(n-1)$-leaf subtrees.

**Theorem A.7.** *Let $\mathcal{T}$ be an unrooted $n$-leaf tree with leaves $N = \{1, \ldots, n\}$ and corresponding leaf weights $\alpha_1, \ldots, \alpha_n$. Similarly, label the internal edges*

*$I_1, \ldots, I_{n-3}$ with edge weights $\alpha_{I_1}, \ldots, \alpha_{I_{n-3}}$. If the Shapley values for all $(n-1)$-leaf subtrees are known, then the Shapley value for $N$ is*

$$\varphi(N, v_{\mathcal{T}}) = \frac{1}{n}\left(\vec{\ell} + \sum_{\substack{S \subseteq N \\ |S|=n-1}} \varphi(N, v_{S,\mathcal{T}})\right), \tag{A.2}$$

*where $\vec{\ell}$ is the vector of leaf weights* $\begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix}$.

*Proof.* First we will show this theorem is true when calculating the Shapley value for one leaf. Fix $i \in \{1, \ldots, n\}$. Using Theorem A.3,

$$\varphi_i(N, v_{\mathcal{T}}) = \frac{1}{n}\left((n-1)\alpha_i + \sum_{\substack{j \in \{1,\ldots,n\} \\ i \neq j}} \frac{1}{n-1}\alpha_j + \sum_{k=1}^{n-3} \frac{f_N(i,k)}{c_N(i,k)}\alpha_{I_k}\right)$$

$$= \frac{1}{n}\left(\alpha_i + \frac{1}{n-1}\left((n-1)(n-2)\alpha_i + \sum_{\substack{j \in \{1,\ldots,n\} \\ i \neq j}} \frac{n-2}{n-2}\alpha_j + (n-1)\sum_{k=1}^{n-3} \frac{f_N(i,k)}{c_N(i,k)}\alpha_{I_k}\right)\right) \tag{A.3}$$

where $f_N, c_N$ are determined with respect to $\mathcal{T}$.

Let $T'$ be the subtree of $\mathcal{T}$ spanned by $N \setminus \{x\}$ where $x \neq i$. Again, using Theorem A.3,

$$\varphi_i(N \setminus \{x\}, v_{T'}) = \frac{1}{n-1}\left((n-2)\alpha_i + \sum_{\substack{j \in \{1,\ldots,n\} \\ i \neq j \neq x}} \frac{1}{n-2}\alpha_j + \sum_{k=1}^{n-3} \frac{f_{N\setminus\{x\}}(i,k)}{c_{N\setminus\{x\}}(i,k)}\alpha_{I_k}\right)$$

where $f_{N\setminus\{x\}}, c_{N\setminus\{x\}}$ are determined with respect to $T'$.

We can see that $i$ is a member of $n-1$ of the $(n-1)$-leaf subtrees and every other leaf is in $n-2$ of those subtrees. Using these facts and Lemma A.6, we can rewrite (A.3) as

$$\frac{1}{n}\left(\alpha_i + \sum_{\substack{i \in S \subset N \\ |S|=n-1}} \varphi_i(N, v_S)\right).$$

Therefore,

$$\varphi(N, v_{\mathcal{T}}) = \frac{1}{n}\left(\vec{\ell} + \sum_{\substack{S \subseteq N \\ |S|=n-1}} \varphi(N, v_S)\right).$$

□

*Example* A.8. Consider the five-leaf tree $\mathcal{T}$ in figure A.5. From direct calculations we see that

$$\varphi(\{A, B, C, D, E\}, v_{\mathcal{T}}) = (5.28, 6.78, 4.2, 4.95, 2.78).$$

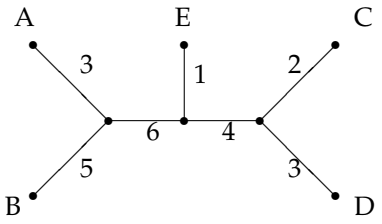



Figure A.5: Example of calculating the Shapley value of a five-leaf tree from all 4-leaf subtrees.

The Shapley value for each four-leaf subtree is

$$\begin{aligned}
\varphi(N, v_{\{A,B,C,D\}}) &= (5.58, 6.92, 4.92, 5.58, 0),\\
\varphi(N, v_{\{A,B,C,E\}}) &= (4.75, 6.08, 6.75, 0, 3.42),\\
\varphi(N, v_{\{A,B,D,E\}}) &= (4.83, 6.17, 0, 7.5, 3.5),\\
\varphi(N, v_{\{A,C,D,E\}}) &= (8.25, 0, 3.58, 4, 25, 2.92),\\
\varphi(N, v_{\{B,C,D,E\}}) &= (0, 9.75, 3.75, 4.42, 3.08).
\end{aligned}$$

Using (A.2), we get

$$\begin{aligned}\varphi(\{A,B,C,D,E\},v_T) &= \frac{1}{5}\left(\begin{pmatrix}3\\5\\2\\3\\1\end{pmatrix} + \sum_{\substack{S\subseteq N\\|S|=n-1}} \varphi(N,v_S)\right)\\ &= \frac{1}{5}\left(\begin{pmatrix}3\\5\\2\\3\\1\end{pmatrix} + \begin{pmatrix}23.41\\28.92\\19\\21.75\\12.92\end{pmatrix}\right)\\ &= (5.28, 6.78, 4.2, 4.95, 2.78).\end{aligned}$$

### A.4.3 Generalizing Theorem A.7

Now that we have looked at calculating the Shapley value from the $(n-1)$-leaf subtrees of a game tree, it would be nice to generalize the formula so we can use any size subtrees. Although it looks as if it would be easy to induct on (A.2), it is a bit tricky when it comes to figuring out what the entries of $\vec{\ell}$ should be. In some cases, the $i$th entry of $\vec{\ell}$ will be a sum of internal edge weights with $i$'s leaf weight. The following example illustrates this situation.

*Example* A.9. In the case of a five-leaf tree, we can calculate the Shapley value for $A$ from (A.2) by

$$\varphi_A(N,v_T) = \frac{1}{5}(\alpha + \sum_{\substack{S\subseteq N\\|S|=4}} \varphi_A(N,v_S)).$$

If we want to calculate the Shapley value for $A$ from the three-leaf subtrees

we obtain

$$\begin{aligned}\varphi_A(N, v_{\mathcal{T}}) &= \frac{1}{5}\left(\alpha + \sum_{\substack{S\subseteq N\\|S|=4}} \frac{1}{4}(\alpha' + \sum_{\substack{U\subseteq S\\|U|=3}} \varphi_A(N, v_U))\right)\\ &= \frac{1}{5}\left(\alpha + \frac{1}{4}\left(4\alpha + \mu + 2\sum_{\substack{U\subseteq S\\|U|=3}} \varphi_A(N, v_U)\right)\right)\\ &= \frac{1}{5}\left(2\alpha + \frac{1}{4}\left(\mu + 2\sum_{\substack{U\subseteq S\\|U|=3}} \varphi_A(N, v_U)\right)\right).\end{aligned}$$

The summand $\frac{1}{4}\mu$ came from the factor of $\vec{\ell}$ from the subtree $ACDE$. See figure A.6. In this case, the leaf weight $\alpha'$ of $A$ is $\alpha + \mu$.

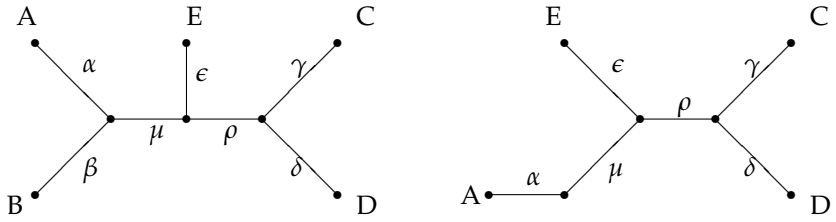



Figure A.6: (*left*) The five-leaf tree where the players are A, B, C, D, and E. (*right*) The four-leaf subtree ACDE. Notice that the leaf weight of A is now $\alpha + \mu$ instead of just $\alpha$.

Taking the internal edge weights into account, we get an equation for the Shapley value from $(n - m)$-leaf subtrees.

**Theorem A.10.** *Let $N = \{1, \ldots, n\}$ be the leaves of tree $\mathcal{T}$ and label the internal edges $\{n + 1, \ldots, 2n - 3\}$. Let the associated edge weights be $\alpha_k$ for $k \in \{1, \ldots, 2n - 3\}$. If all of the Shapley values for the $(n - m)$-leaf subtrees are known, then the Shapley value of $\mathcal{T}$ is*

$$\varphi(N, v_{\mathcal{T}}) = \frac{1}{n!}\left(\vec{L}(m) + (n - m)!m! \sum_{\substack{S\subseteq N\\|S|=n-m}} \varphi(N, v_S)\right) \tag{A.4}$$

*where*

$$\vec{L}_i(m) = \sum_{k=1}^{2n-3} \sum_{j=c(i,k)}^{m} (n-j)!(j-1)! \binom{f(i,k)}{n-j} \alpha_k. \tag{A.5}$$

*Proof.* We will prove this by induction on $m$. It suffices to prove this for a single leaf so fix $i \in N$.

Base case: When $m = 1$, we have

$$\frac{1}{n!} \left( \vec{L}_i(1) + (n-1)!1! \sum_{\substack{i \in S \subseteq N \\ |S|=n-1}} \varphi_i(N, v_S) \right)$$

$$= \frac{1}{n!} \left( (n-1)!(1-1)! \binom{1}{n-1} \alpha_i + (n-1)! \sum_{\substack{i \in S \subseteq N \\ |S|=n-1}} \varphi_i(N, v_S) \right)$$

$$= \frac{1}{n} \left( \alpha_i + \sum_{\substack{i \in S \subseteq N \\ |S|=n-1}} \varphi_i(N, v_S) \right).$$

By theorem A.7 this is $\varphi_i(N, v_{\mathcal{T}})$.

Induction Hypothesis:

$$\varphi_i(N, v_{\mathcal{T}}) = \frac{1}{n!} \left( \vec{L}_i(m-1) + (n-m+1)!(m-1)! \sum_{\substack{i \in S \subseteq N \\ |S|=n-m+1}} \varphi_i(N, v_S) \right).$$

Inductive Step: We can apply theorem A.7 to the induction hypothesis to get

$$\varphi_i(N, v_{\mathcal{T}}) = \frac{1}{n!} \left( \vec{L}_i(m-1) + (n-m)!(m-1)! \sum_{\substack{S \subseteq N \\ |S|=n-m+1}} \left( \alpha'_i + \sum_{\substack{i \in U \subseteq N \\ |U|=n-m}} \varphi_i(N, v_U) \right) \right) \tag{A.6}$$

where $\alpha'_i$ is the leaf weight of $i$ in the subtree spanned by $U$. Notice that any edge weight is in $\alpha'_i$ if $U \setminus \{i\}$ is from the opposite side of that edge from $i$. Thus

$$(n-m)!(m-1)! \sum_{\substack{S \subseteq N \\ |S|=n-m+1}} \alpha'_i = \sum_{k=1}^{2n-3} (n-m)!(m-1)! \binom{f(i,k)}{n-m} \alpha_k.$$

If we add this to $\vec{L}_i(m-1)$ we get $\vec{L}_i(m)$. Also note that every $(n-m)$-leaf subtree comes from $m$ $(n-m+1)$-leaf trees so plugging that into (A.6) yields

$$\varphi_i(N, v_{\mathcal{T}}) = \frac{1}{n!}\left(\vec{L}_i(m) + (n-m)!m! \sum_{\substack{i \in U \subset N \\ |U|=n-m}} \varphi_i(N, v_U)\right).$$

□

It is interesting to note that (A.4) does not seem to depend on the topology of the tree so we can theoretically induct on the size of the subtrees. However, with the addition of (A.5) we lose the ability to induct since we need to know the topology of the tree or the split counts. Equation (A.4) is helpful in seeing how the Shapley value depends on the each of the edge weights. We can see this from the following corollary.

**Corollary A.11.** *If $N$ is the set of leaves in tree $\mathcal{T}$ with edge weights $\alpha_k$ for $k = 1, \ldots, 2n-3$, then for $i \in N$,*

$$\varphi_i(N, v_{\mathcal{T}}) = \frac{1}{n!}\left(\sum_{k=1}^{2n-3} \sum_{j=c(i,k)}^{n-2} (n-j)!(j-1)! \binom{f(i,k)}{n-j} \alpha_k\right).$$

*Proof.* Use theorem A.10 when $m = n-2$ and

$$\sum_{\substack{S \subset N \\ |S|=2}} \varphi_i(S, v_{\mathcal{T}}) = \sum_{k=1}^{2n-3} \frac{1}{2} \alpha_k f(i,k).$$

□

### A.4.4 Examining the Null Space

As we have seen, Theorem A.3 has been instrumental in showing how the Shapley value can be calculated from the Shapley value of any size subtrees. Now we will also use this theorem to understand the dependence of the null space on the split counts, as mentioned in section A.3.4.

The following theorem exhibits the null spaces basis of $\mathbf{M}$ in terms of the split counts.

**Theorem A.12.** *Let $\mathcal{T}$ be an $n$-leaf tree with leaves $N = \{1, \ldots, n\}$ and internal edges $I_1, \ldots, I_{n-3}$. For each internal edge $I_k$, there corresponds a vector $w_{I_k} \in \mathbb{R}^{2n-3}$ in a basis of the null space of the Shapley transformation of $\mathcal{T}$:*

$$(w_{I_k})_i = \begin{cases} -\frac{f(i,k)-1}{(n-2)c(i,k)} & \text{if } 1 \leq i \leq n \\ 1 & \text{if } i = n+k \\ 0 & \text{otherwise} \end{cases} \tag{A.7}$$

*for all $k \in \{1, \ldots, n-3\}$ and entries $i \in \{1, \ldots, 2n-3\}$, where the first $n$ entries correspond to leaves and the last $n-3$ entries corresponds to internal edges.*

Before proving the theorem, we give an example.

*Example* A.13. Consider the five-leaf tree in Figure A.3. Label the internal edges $I_1, I_2$ such that the corresponding edge weights are $\mu, \rho$, respectively. Using Theorem A.12, let us calculate the null space vector $w_{I_1}$. We know that the $5+1 = 6$th entry of $w_{I_1}$ is 1 and all entries after that are zero. To find the first five entries of the vector, we consider the two subtrees obtained by removing $I_1$ from the tree. In that case, we'll get the subtrees AB and CDE. Then using (A.7), the first two entries of the matrix corresponding to A and B will be

$$-\frac{3-1}{(5-2)2} = -\frac{1}{3}$$

and the next three entries corresponding to C, D, and E are

$$-\frac{2-1}{(5-2)3} = -\frac{1}{9}.$$

These values correspond to the first vector in the null space basis we presented in Section A.3.2. We may obtain the other basis vector in a similar fashion, by considering edge $I_2$ instead of $I_1$.

Now we will prove Theorem A.12.

*Proof.* Let $\mathcal{T}$ be an $n$-leaf tree. Consider the $i$th leaf. If we let $M$ be the matrix of Shapley value coefficients for $\mathcal{T}$ then we want to show

$$\sum_{j=1}^{2n-3} \mathbf{M}[i,j](w_{I_k})_j = 0. \tag{A.8}$$

Fix $k \in \{1, \ldots, n-3\}$. There are a couple of notes to point out that make this proof easier. First using Theorem A.3, for all leaves $j \neq i$,

$$\mathbf{M}[i,j] = (n-2)!$$

and

$$\mathbf{M}[i,i] = (n-1)(n-1)!$$

for $j \in \{1,\ldots,n\}$ (not including the factor of $\frac{1}{n!}$). The only other entry of the matrix we need to consider is $M[i,n+k]$ since our construction of $w_{I_k}$ has zeros for the rest of the entries. Thus

$$\mathbf{M}[i,n+k] = (n-1)!\frac{f(i,n+k)}{c(i,n+k)}.$$

Plugging all of this into (A.8) yields

$$\sum_{j=1}^{2n-3} \mathbf{M}[i,j](w_{I_k})_j = -f(n-2)!\frac{c-1}{(n-2)f} - (c-1)(n-2)!\frac{f-1}{(n-2)c} - (n-1)(n-1)!\frac{f-1}{(n-2)c} + (n-1)!\frac{f}{c}.$$

To show this is the same as showing

$$(n-1)!\frac{f}{c} = f(n-2)!\frac{c-1}{(n-2)f} + (c-1)(n-2)!\frac{f-1}{(n-2)c} + (n-1)(n-1)!\frac{f-1}{(n-2)c}. \tag{A.9}$$

The right side of the equation (A.9) is

$$\begin{aligned}
&f(n-2)!\frac{c-1}{(n-2)f} + (c-1)(n-2)!\frac{f-1}{(n-2)c} + (n-1)(n-1)!\frac{f-1}{(n-2)c} \\
&= (n-2)!\left(\frac{c-1+f-1}{n-2}\right) + (n-1)(n-1)!\frac{f-1}{(n-2)c} - (n-2)!\frac{f-1}{(n-2)c} \\
&= (n-2)!\left(\frac{(c-1)(n-1)}{(n-2)c}\right) + (n-1)(n-1)!\frac{f-1}{(n-2)c} \\
&= \frac{(n-1)!}{(n-2)c}(f(n-2)) \\
&= (n-1)!\frac{f}{c}.
\end{aligned}$$

Thus $w_{I_k}$ is in the null space of the Shapley value. It is apparent that the null space has dimension $n-3$ and the $w_{I_k}$ are linearly independent. Therefore the $w_{I_k}$ form a basis of the null space of $\mathbf{M}$. □

This theorem suggests that one may determine the topology of the tree from the null space $Null(\mathbf{M})$ of its Shapley transformation $\mathbf{M}$. Because every different $n$-leaf tree topology divides the leaves differently with respect to at least one leaf (hence producing a different split count), the null space bases will differ in at least one vector. Thus $Null(\mathbf{M})$ will distinguish the correct tree topology.

An immediate corollary is that $Null(\mathbf{M})$ reveals the location of *cherries*. A pair of leaves $(i, j)$ is called a *cherry* if they have a common parent. This is the case if and only if the tree spanned by $i$ and $j$ does not include an internal edge. Therefore, removing the internal edge that contains the common parent splits $\mathcal{T}$ into a 2-leaf and an $(n-2)$-leaf subtree. Using Theorem A.12 which determines a specific basis for the nullspace, we may detect which edges include the parent of a cherry. This may be verified in the previous examples.

**Corollary A.14.** *Let $\mathcal{T}$ be an unrooted tree with leaves set $N$ and edge set $E$. Let $w^k := w_{I_k} = (w_1^k, \ldots, w_n^k, w_{n+1}^k, \ldots, w_{2n-3}^k)$ denote the basis vectors of the nullspace of $\varphi(v_{\mathcal{T}})$. Then there is a tree $\mathcal{T}'$ with same leaf set $\varphi(v_{\mathcal{T}}) = \varphi(v_{\mathcal{T}'})$ in which the pair $(i, j)$ of leaves form a cherry if and only if there exists $k'$ such that*

$$w_i^{k'} = w_j^{k'} = -\frac{n-3}{2(n-2)}. \tag{A.10}$$

*Proof.* Inspecting (A.7) in Theorem A.12 reveals the equivalence, since in case (and only in case) that deletion of $I_{k'}$ splits the tree into one with two and one with $n-2$ leaves, the above stated entries in $w^{k'}$ prevail. □

## A.5 Characterization of the Shapley Value of Tree Games

The axioms presented in Section 2 uniquely characterize the Shapley value on the class of all $n$-person games. However, the class of $n$-person games that are derived from a tree is much smaller. By $\mathcal{V}^{N,E}$ we denote the class of games arising from some tree with set of leaves $N$ and edge set $E$. For games in $\mathcal{V}^{N,E}$ we will allow positive as well as non-positive edge weights. Thus, $\mathcal{V}^{N,E}$ is a linear space and we ask for its dimension.

For a fixed pair $(N, E)$ define games $v_k$ $(k \in E)$ in the following way: $v_k$ corresponds to the tree in which edge $k$ is weighted 1 and all other edges are weighted zero. We call such a game a *basis game*. It is readily checked that the game $v$ associated with the tree that exhibits edge weights $\alpha_1, \ldots, \alpha_n, \alpha_{I_1}, \ldots, \alpha_{I_{n-3}}$ is the linear combination $v = \sum_{k \in E} \alpha_k v_k$. Moreover,

the family $(v_k)_{k\in E}$ is linearly independent. Therefore these games form a basis of $\mathcal{V}^{N,E}$ and dim $\mathcal{V}^{N,E} = 2n - 3$.

Next, we examine a basis game $v_k$ and ask for a "reasonable" distribution $\psi(v_k) \in \mathbb{R}^n$. The total diversity is $v_k(N) = 1$. We may interpret zero edge weights on either side of the edge $k$ as having two groups of species, each one being homogeneous. So a natural property would be that the degree of diversity that we assign to one group does only depend on the fraction of this group (and hence of the fraction of the other group) relative to the whole population. It seems plausible that a group on one side of the edge (relatively) diversifies the population more, the more species there are on the other side of the edge. Thus, we may assume that $\psi_i(v_k)$ is described by a function that is increasing in the fraction $f(i,k)/n$. We formulate these considerations as an additional axiom.

**Axiom (group proportionality on basis games):** For fixed $N$ and $E$, a mapping $\psi : \mathcal{V}^{N,E}$ is said to satisfy *group proportionality on basis games*, if there is some constant $d \in \mathbb{R}$ such that $\psi$ satisfies $\sum_{i\in\mathcal{C}(i,k)} \psi_i(v_k) = d\,\frac{f(i,k)}{n}$ for all $i \in N, k \in E$.

Thus, with $\psi$ satisfying this axiom, a groups assigned diversity linearly changes with the other group's fraction of the whole population. Using the new axiom, we get a characterization result on $V^{N,E}$.

**Theorem A.15.** *For each pair $(N,E)$ (consisting of leaf set $N$ and edge set $E$) there is one and only one mapping $\psi : \mathcal{V} \to \mathbb{R}^n$ that satisfies Pareto efficiency, symmetry, additivity and group proportionality. This mapping coincides with the Shapley value, i.e., $\psi = \varphi$.*

*Proof.* It is immediately verified that the Shapley value satisfies all the axioms (for group proportionality use A.3).
Now, let $(N,E)$ be fixed and $\psi$ satisfy the axioms. First, we take a basis game $v_k$ and determine $\psi$. By symmetry, we may conclude $\sum_{i\in\mathcal{C}(i,k)} \psi_i(v_k) = c(i,k)\,\psi_i(k) = c(j,k)\,\psi_j(v_k)$ for $i,j \in \mathcal{C}(i,k) = \mathcal{C}(j,k)$. Pareto efficiency implies $v_k(N) = 1 = \sum_{j\in N} \psi_j(v_k) = \sum_{j\in\mathcal{C}(i,k)} \psi_j(v_k) + \sum_{j\in\mathcal{F}(i,k)} \psi_j(v_k) = d\,(\frac{f(i,k)}{f(i,k)+c(i,k)} + \frac{c(i,k)}{f(i,k)+c(i,k)}) = d$. Hence, we obtain $\psi_i(v_k) = \frac{f(i,k)}{n\,c(i,k)}$ for any $i \in N$ and $k \in E$. Analogously, we get $\psi_i(\lambda v_k) = \lambda\,\psi_i(v_k)$ for $\lambda \in \mathbb{R}$. Using additivity and Theorem A.3, $\psi$ coincides with the Shapley value on $\mathcal{V}^{N,E}$. □

We close this section with two remarks. First, note that any game arising from a tree with nonnegative edge weights is representable as a linear

combination of basis games using nonnegative coefficients. Hence, we may derive a version of Theorem A.15 for classes of games that actually arise from phylogenetic trees.

Second, Theorem A.15 provides further justification for the use of the Shapley value to analyze phylogenetic trees. If one wants to distribute the total diversity of a population on its species and the distribution rule should satisfy the above (reasonable) axioms, then the Shapley value is the only possible choice. As symmetry, Pareto efficiency and additivity are rather "obligatory" requirements for a plausible rule, it is the proportionality axiom that provides further insight in the rationale behind the Shapley value. Of course, modification of the group proportionality axiom eventually leads to a different distribution rule based on a different rationale.

## A.6 The Core of Tree Games

Thus far we have been using the Shapley value to solve tree games. However, another solution concept for $n$-player cooperative games that is frequently studied is the *core* of a game, which is the set of all imputations $\vec{x} \in \mathbb{R}^n$ such that for all coalitions $S \subseteq N$, $\sum_{i \in S} x_i \geq v(S)$. In this section we examine the core of phylogenetic tree games.

It is apparent that the core for a single player game is 0 and the core of the two player game is $\{(x_1, x_2) \in \mathbb{R}^2 | x_1 + x_2 = \alpha, x_1 \geq 0, x_2 \geq 0\}$ so we will derive the core for the three- and four-leaf tree games to gain some intuition about what the core looks like.

*Example* A.16. The characteristic function of the three-leaf tree is given in section A.3.1. From this we get the following system of inequalities:

$$\begin{aligned} x_A + x_B + x_C &= \alpha + \beta + \gamma \\ x_A + x_B &\geq \alpha + \beta \\ x_A + x_C &\geq \alpha + \gamma \\ x_B + x_C &\geq \beta + \gamma \end{aligned}$$

It is apparent that the core consists of the single element $\vec{\ell}$ which is the leaf weights $\begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix}$.

So we see that the three-leaf tree has only one element in its core, namely the vector of leaf weights. Now we will look at the four-leaf tree game which will help us see how internal edges affect the core.

*Example* A.17. It is easy enough to derive the characteristic function of the four-leaf tree game given in figure A.3 so we will not write it here. This game yields the following system of inequalities:

$$\begin{aligned} x_A + x_B + x_C + x_D &= \alpha + \beta + \mu + \gamma + \delta \\ x_A + x_C &\geq \alpha + \mu + \gamma \qquad \text{(A.11)} \\ x_B + x_D &\geq \beta + \mu + \delta \qquad \text{(A.12)} \\ &\vdots \end{aligned}$$

From (A.11) and (A.12) we see that

$$\alpha + \mu + \gamma \leq x_A + x_C \leq \alpha + \gamma.$$

So either $\mu = 0$ in which case we have a degenerate tree and the core is $\vec{\ell}$ or the core has to be empty since the inequality cannot be satisfied.

From these two examples we obtain the following theorem.

**Theorem A.18.** *Let $\mathcal{T}$ be an $n$-leaf game tree $\mathcal{T}$ where $n \geq 3$. If the tree is degenerate, then the core consists of the leaf weight vector $\vec{\ell}$. Otherwise the core is empty.*

*Proof.* Let $\mathcal{T}$ be an $n$-leaf tree with edge weights $\alpha_i$ for $i \in \{1, \ldots, 2n-3\}$. Every tree has at least two cherries, where a *cherry* is a set of two leaves with a common parent. Label the two leaves on one cherry 1 and 2 and label the two leaves on the other cherry 3 and 4 each with corresponding leaf weights $\alpha_1$, $\alpha_2$, $\alpha_3$ and $\alpha_4$. We know from the properties of the core that for the set of leaves $N$,

$$\sum_{j \in N} x_j = \sum_{i \in \{1,\ldots,2n-3\}} \alpha_i \qquad \text{(A.13)}$$

$$x_1 + x_3 \geq \sum_{k \in P} \alpha_k \qquad \text{(A.14)}$$

$$\sum_{j \in N \setminus \{1,3\}} x_j = \sum_{i \in T \setminus P} \alpha_i \qquad \text{(A.15)}$$

where $P$ is the set of edges in the subtree spanned by A and C. From (A.13) and (A.15) we get

$$x_1 + x_3 \leq \alpha_1 + \alpha_3. \qquad \text{(A.16)}$$

We know there are no other edge weights included in (A.16) because the subtree spanned by 2 and 4 (which is included in $T \setminus P$) will have the same

edges as $P$ except for the leaf weights. Thus from (A.14) and (A.16) we must have

$$\sum_{k \in P} \alpha_k \leq x_1 + x_3 \leq \alpha_1 + \alpha_3.$$

However this cannot be satisfied and the core is empty unless all of the internal edge weights are zero (i.e., the tree is *degenerate*), in which case the core is the element $\vec{\ell}$.

□

Notice that for $n = 3$, $\mathcal{T}$ is always degenerate, and thus the core will never be empty.

Because the core of tree games is empty in most cases, the Shapley value is a far more interesting solution concept to consider. However, the core has the potential to find (or rule out) degenerate trees easily, unlike the Shapley value.

Suppose we are given the pairwise distances for $n$ leaves of a tree. If any four leaf subset has an empty core, then the tree is definitely not degenerate. But if any of the inequalities hold then the subtree spanned by the four leaves in the subset contains a degeneracy. To illustrate this point, see example A.19.

*Example* A.19. Consider the 5-leaf tree given in figure A.3. Let $\mu > 0$ and $\rho = 0$. Then the four-leaf subtree ACDE has a nonempty core, namely $\begin{pmatrix} \alpha + \mu \\ 0 \\ \gamma \\ \delta \\ \epsilon \end{pmatrix}$. Thus there is a degeneracy among the leaves ACDE which we can see (C, D, E all have a common parent). However, in the four leaf subtree ABCE, we have

$$\alpha + \mu + \gamma \leq x_A + x_C \leq \alpha + \gamma$$

so the core is empty. Thus the tree is not totally degenerate but it contains a degenerate subtree CDE.

## A.7 Conclusion

In this paper we have presented a biological interpretation of the Shapley value on games derived from phylogenetic trees. From a mathematical perspective, we showed how the Shapley value of tree games can be calculated

from the Shapley value of the subtrees even if the tree itself cannot be constructed from those subtrees. It is worth noting again the dependence many of our results have on the *split counts*, the division of leaves with respect to a given edge. We have also proved some results about the null space of the Shapley transformation on tree games, as well as the emptiness of the core.

Our work suggests several directions for further research. For instance:

- Can our results be used in some way to assist with reconstruction of trees from data?
- Is there a way to determine split counts from raw data, and can this assist in determining the correct tree topology?
- If there were a way to estimate the Shapley value from data, this would be enough to determine edge weights of a degenerate tree. Do the leaf weights of this tree have any significance?
- If we use the Shapley value to rank the species in the Noah's ark problem for preservation, to what extent can we guarantee that the diversity of the top $k$ species (i.e., the weight of the subtree spanning them) approximates the total diversity of all $n$ species? Determine a bound that depends on $k$ and $n$.

# Appendix B

# Poster: "The Shapley Value of Phylogenetic Trees"

This is a copy of the poster I presented at the sixth annual Harvey Mudd College mathematics conference titled "Algebra, Geometry, and Phylogenetic Trees" held on October 23, 2004. I also used this poster at the AMS-MAA-SIAM Joint conference at Atlanta in January 2005.

# The Shapley Value of Phylogenetic Trees

**Claus-Jochen Haake[1], Akemi Kashiwada[2,*], Francis Edward Su[3,*]**

*[1]University of Bielefeld, [2,3]Harvey Mudd College*

## Introduction

- A phylogenetic tree is a weighted binary tree.
- A *cooperative game (N,v)* is a set of players $N$ and a "worth" function $v: 2^N \to \mathbf{R}$.
- The *Shapley value* $\varphi(N,v)$ determines the average worth of each player and is one of the most important concepts in game theory.
- We define a *tree game* $v_T$ to be a cooperative game where the worth of a coalition is the weight of the subtree spanned by its members.

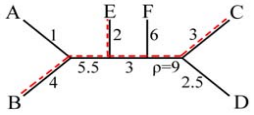



Figure 1: A 6-leaf tree with the subtree spanned by *BCE* in red.

**Example:** We can define a tree game $v_T$ on the players $N=\{A,B,C,D,E,F\}$. The worth of coalition *BCE* is

$v_T(BCE) = 4+5.5+3+9+3+2 = 26.5$

The Shapley value is

$\varphi(\{A,B,C,D,E,F\}, v_T) = (4.5,6.9,6.975,6.575,3.925,7.125)$.

## Interpretation

We can think of the Shapley value of species *i* as the average diversity *i* contributes to any group. The Shapley value satisfies the following axioms on tree games:

1. Pareto efficiency
2. Symmetry
3. Dummy Axiom
4. Additivity
5. Group proportionality

**Theorem** (Haake-Kashiwada-Su): The Shapley value is the unique solution satisfying the above axioms.

Shapley (1953) showed that (1)-(4) characterize the Shapley value on all games. We showed that for the smaller class of tree games an additional axiom (5) is needed.

## Entries of the Shapley Value Matrix

- The Shapley value linearly depends on the edge weights $\vec{E}$ so there is an $n \times 2n\text{-}3$ matrix $M$ such that

$$\varphi(N, v_T) = M\vec{E}$$

- Define $f(i,k)$ to be the number of leaves on opposite side of leaf $i$ from edge $k$ (the "far" side) and $c(i,k)$ the number of leaves on the same side of edge $k$ as leaf $i$ (the "close" side).

**Example:** In Figure 1, $f(A,\rho) = 2$, $c(A,\rho) = 4$.

**Theorem** (Haake-Kashiwada-Su): The value of the $(i,k)$-entry of $M$ is $f(i,k)/nc(i,k)$.

**Example:** The Shapley value matrix for the six-leaf tree in Figure 1 is shown on the right. Verify the entry of the contribution of $\rho$ to the Shapley value of $A$ is

$$M[1,8] = \frac{2}{4(6)} = \frac{1}{12}$$

$$M = \frac{1}{720}\begin{bmatrix} 600 & 24 & 24 & 24 & 24 & 24 & 240 & 60 & 120 \\ 24 & 600 & 24 & 24 & 24 & 24 & 240 & 60 & 120 \\ 24 & 24 & 600 & 24 & 24 & 24 & 60 & 240 & 120 \\ 24 & 24 & 24 & 600 & 24 & 24 & 60 & 240 & 120 \\ 24 & 24 & 24 & 24 & 600 & 24 & 60 & 60 & 120 \\ 24 & 24 & 24 & 24 & 24 & 600 & 60 & 60 & 120 \end{bmatrix}$$

## Shapley Value from Subtrees

**Theorem** (Haake-Kashiwada-Su): The Shapley value of an $n$-leaf tree $T$ with leaf weights $\vec{L}$ can be calculated from the Shapley value of all $(n\text{-}1)$-leaf subtrees $S$ of $T$ by

$$\varphi(N, v_T) = \frac{1}{n}\left(\vec{L} + \sum_{S \subset N} \varphi(N, v_S)\right)$$

By contrast, a tree cannot be reconstructed from its $(n\text{-}1)$-leaf subtree weights (Pachter-Speyer 2004).

**Example:** In the 5-leaf tree in Figure 2, $\vec{E} = (3,5,2,3,1,6,4)^T$. From direct calculations we know the Shapley value of each 4-leaf subtree so

$$\varphi(\{A,B,C,D,E\}, v_T) = \frac{1}{5}\left((3,52,3,1) + \sum_{S \subset N} \varphi(N, v_S)\right) = \frac{1}{5}\left((3,5,2,3,1) + (23.41,28.92,19,21.75,12.92)\right)$$

$$= (5.28,6.78,4.2,4.95,2.78)$$

**Theorem** (Haake-Kashiwada-Su): The Shapley value of an $n$-leaf tree $T$ can be calculated from the Shapley value of all $(n\text{-}m)$-leaf subtrees of $T$ by

$$\varphi(N, v_T) = \frac{1}{n!}\left(\vec{L}(m) + (n-m)!m! \sum_{S \subset N} \varphi(N, v_S)\right)$$

$$\vec{L}(m) = \sum_{k=1}^{2n-3} \sum_{j=b(i,k)}^{m} (n-j)!(j-1)!\left\{ {t(i,k) \atop n-j} \right\} \alpha_k$$

## Null Space

**Theorem** (Haake-Kashiwada-Su): The $n\text{-}3$ vectors $w_{Ik}$ form a basis of $M$, one for each internal edge $I_k$:

$$(w_{I_k})_i = \begin{cases} -\dfrac{f(i,k)-1}{(n-2)c(i,k)} & \text{if } i \text{ is a leaf} \\ 1 & \text{if } i = n+k \\ 0 & \text{otherwise} \end{cases}$$

Thus, different tree topologies yield different null spaces.

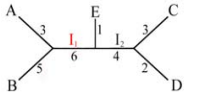



Figure 2: A 5-leaf tree. We are calculating the null space basis vector associated with $I_1$.

**Example:**
A null space basis for $M$ is

$$\left\{\left(-\frac{1}{3},-\frac{1}{3},-\frac{1}{9},-\frac{1}{9},-\frac{1}{9},1,0\right)^T, \left(-\frac{1}{9},-\frac{1}{9},-\frac{1}{3},-\frac{1}{3},-\frac{1}{9},0,1\right)^T\right\}$$

The first entry of the first basis vector was obtained by noting $f(A,I_1) = 3$ and $c(A,I_1) = 2$. Then $-(3\text{-}1)/(5\text{-}2)(2) = -1/3$.

**Further Question**

Can the null space of $M$ be determined from data? If so, this would yield a new method for reconstructing trees.

## The Core

**Theorem** (Haake-Kashiwada-Su): The core of tree games is empty unless the tree is degenerate.

*We would like to acknowledge partial support from NSF Grant DMS-0301129 and a Howard Hughes Medical Institute grant.

For more information, contact akashiwada@hmc.edu

Preprint at: http://www.math.hmc.edu/~su/papers.html

# Appendix C

# Poster: "Reconstructing Phylogenetic Trees from Subsplits"

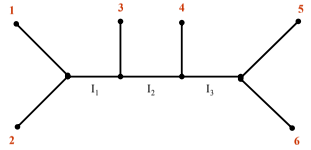
This is a copy of the poster I made for Harvey Mudd College's 2005 Presentation Days on my thesis work.

# Reconstructing Phylogenetic Trees from Subsplits

Akemi Kashiwada*, Professor Francis Su (Advisor), Professor Stephen Adolph (Second Reader)

## Introduction

One of the major problems in mathematical biology is the reconstruction of phylogenetic trees. In 1971, Peter Buneman addressed this problem by showing that any binary tree can be uniquely reconstructed from its *splits*, denoted $A|B$ which are bipartitions of the leaf set. [1] We can gain some insight to this idea by noting that each split corresponds to an edge of the tree that disconnects the sides of the partition of leaves.




The nontrivial splits of this 6-leaf tree are

**12|3456, 123|456, 1234|56**

which correspond to edges $I_1$, $I_2$, and $I_3$, respectively.

## Subsplits

• We define an *(n-1)-leaf subsplit* to be bipartition of a *(n-1)*-leaf subset of the leaves.
• Two subsplits $A|B$ and $C|D$ are *compatible* if at least one of the following is empty:

$$A\cap C,\ A\cap D,\ B\cap C,\ B\cap D$$

• We call these empty intersections *connections*, denoted $\{A,C\}$.
• If $C$ contains the missing leaf of $A|B$, then we say that connection is *constraining* for $A|B$ and denote it $\{A,\underline{C}\}$.

For example, a set of 5-leaf subsplits of the 6-leaf tree above is

**12|356, 13|456, 123|46**

With connections

{12, 456}, {12, 46}, {13, 46}, {456, 123}

## Motivation

Unlike the case with splits, reconstruction of an $n$-leaf tree is not guaranteed from a set of compatible *(n-1)*-leaf subsplits. Semple and Steel [2] provide criteria for when a set of subsplits will uniquely construct a phylogenetic tree. Unfortunately one of the criteria requires that we look through all possible trees on that set of leaves and check for whether the tree is consistent with the given subsplits. Since this process is undesirable, **the work we have done provides a constructive method for looking at the trees satisfying this criterion.**

## Tree Reconstruction Algorithm using *(n-1)*-leaf Subsplits

Let $\{S_1, \ldots, S_k\}$ be a set of *(n-1)*-leaf compatible subsplits.

1. Let $C$ be the set of connections between every pair of subsplits with the constraining connections indicated.
2. We will start by adding the missing leaf to all subsplits with at least one constraining connection. Suppose $S_i$ has at least one constraining connection. Then we are left with several cases in which we can add the missing leaf to $S_i$:
   a. If all constraining connections of $S_i$ are associated with the same side of $S_i$ then add the missing leaf to the opposite side. That is, if all constraining connections of $S_i$ are of the form $\{S_i^L, S_j^k\}$, then add the missing leaf to $S_i^R$. Repeat Step 2 with the next subsplit.
   b. If the constraining connections are associated with both sides of $S_i$, then
      i. If $\{S_j\}_{j\in\lambda}$ is the set of all subsplits with a constraining connection associated with $S_i^L$ and there exists another connection $\{S_i^R, \underline{S_x^h}\}$ for all $j\in\lambda$ then:
         add the missing leaf to $S_i^L$
         remove all of the constraining connections of the form $\{S_i^L, \underline{S_x^h}\}$, and
         add the constraint to any connections as necessary.
         Repeat Step 2 with the next subsplit.
      ii. Repeat Step 2bi for the set of constraining connections associated with $S_i^R$.
      iii. We cannot add in the missing leaf and this set of subsplits is unrealizable as an $n$-leaf tree. End the algorithm.
3. If there are any other subsplits left with no constraining connections, then add the missing leaf to either side.

**Main Theorem:** The algorithm will stop after all of the *(n-1)*-leaf subsplits have been transformed into splits if and only if there exists an $n$-leaf tree that is consistent with the subsplits.

## Example

Say we are given the subsplits

**128|3456, 357|2468, 37|12468, 57|12368, 126|3457**

labeled A through E, respectively. We have the connection table

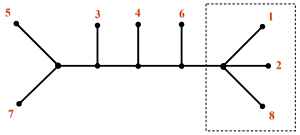
| | L | R | Missing Leaf |
|---|---|---|---|
| A | $\underline{B^L}$ $\underline{C^L}$ $\underline{D^L}$ $\underline{E^R}$ | | 7 |
| B | $\underline{A^L}$ $\underline{C^R}$ $\underline{E^L}$ | $C^L$ $D^L$ | 1 |
| C | $A^L$ $B^R$ $E^L$ | $\underline{B^L}$ $\underline{D^L}$ | 5 |
| D | $A^L$ $\underline{B^R}$ $\underline{C^R}$ $E^L$ | | 4 |
| E | $B^L$ $C^L$ $D^L$ | $\underline{A^L}$ | 8 |

Notice that each subsplit has at least one constraining subsplit we can add the missing leaves in any order.

Since all of the constraining connections are on one side of the split, by Step 2a we just add the missing leaf to the opposite side. Thus the new subsplits are transformed into

**128|34567, 357|12468, 357|12468, 57|123468, 1268|3457**

This corresponds to the following tree:




Notice that there is ambiguity in the tree (in the dashed box on the right) since the second and third subsplits correspond to the same split.

## Example

Consider the 5-leaf subsplits $S_1$, $S_2$, $S_3$ given by

**123|45, 456|23, 234|16**

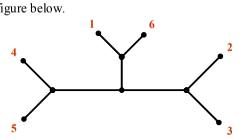
The connections are

{123,456}, {45,23}, {45, 16}, {23, 16}

Since each subsplit has a constraining connection, we start by adding 6 to $S_1$. Notice that 6 is connected to both sides of $S_1$ but $S_3$ has two connections with $S_2$. Since we need to maintain the connection with $S_2$, we need to add 6 to the left-side of $S_1$. This will destroy the connection {123, 456} but the connection {45,23} is still intact. Now to add 1 to $S_2$ we just need to take into consideration the constraining connection {23, 16} so 1 is forced to be on the left-side. Finally, 5 goes on the left-side of $S_3$ resulting in the splits

**1236|45, 1456|23, 2345|16**.

Thus this set of subsplits uniquely constructs the 6-leaf tree in the figure below.




## Non-Example

Suppose we want to construct a 6-leaf tree from the 5-leaf subsplits

**12|345, 16|234, 15|346**

These are compatible with connections

{345, 16}, {12, 346}, {234, 15}

By our algorithm, there is no tree that is consistent with these subsplits which we can also determine by considering the placement of the edges relative to 1 and 6 as dictated by the subsplits.

## References

[1] Buneman, Peter. "The recovery of trees from measures of dissimilarity." In F.R. Hodson, D.G. Kendall, P.T., editor, *Anglo-Romanian Conference on Mathematics in the Archaeological and Historical Sciences*. University Press, 1971; 387-395.
[2] Semple, Charles and Steel, Mike. "A characterization for a set of partial partitions to define an $X$-tree." *Discrete Mathematics* 247 (2002): 169-186.

*I would like to thank the members of the Harvey Mudd Mathematics 2005 Thesis class and faculty for supporting my research

For more information contact akashiwada@g.mud.com
A copy of my thesis can be found at www.math.hmc.edu/~kemikach/thesis

# Bibliography

[1] (2005). Tree of life web project. ¡http://tolweb.org/tree/¿. This website is an ongoing project where biologist from different fields contribute parts of the evolutionary tree in an attempt to construct an all-encompassing phylogenetic tree.

[2] Billera, L. J., Holmes, S. P., and Vogtmann, K. (2001). Geometry of the space of phylogenetic trees. *Advances in Applied Mathematics*, 27(4):733 – 767. Discusses the space of trees. Other papers refer to this one to note that an edge or split corresponds to one of the "axes" of the tree space. General topology of $n$-leaf tree space has moving over an edge correspond to compressing one edge and opening it in another direction.

[3] Böcker, S., Dress, A. W., and Steel, M. (1999). Patching up $x$-trees. *Annals of Combinatorics*, 3:1 – 12. This paper focuses on the unique reconstruction of trees from four-leaf subsplits. Although they claim that it is a simple matter to induct on this work to reproduce results for all other subsplits, I have not found a way to do this without a thorough case analysis.

[4] Böcker, S., Dress, A. W., and Steel, M. (2000). Simple by fundamental limitations on supertree and consensus tree methods. *Systematic Biology*, 49(2):363 – 368. This paper is less mathematically technical and considers reconstruction of trees from subtrees from an axiomatic perspective. They show that their set of reasonable axioms they wish any reconstruction method to possess cannot be simultaneously satisfied by a single method. They state that the purpose of this work is to allow the focus of further reconstruction methods to include some but not all desirable properties.

[5] Bossert, W., Pattanaik, P. K., and Xu, Y. (2003). Similarity of options and the measurement of diversity. *Journal of Theoretical Politics*, 15(4):405 –

421. Rather than looking at specific measures, this article considers axioms that two types of measures of diversity should possess. The types of measures considered are ordinal distances and ratio-scale distances. Axioms for the ordinal distances which are like pairwise distances are monotonicity, indifference between no-diversity situations, and composition consistency, all of which are analogous to the Shapley value axioms.

[6] Bryant, D. (2004). The splits in the neighborhood of a tree. *Annals of Combinatorics*, pages 1 – 11. Uses four different tree metrics and uses splits to determine if a given tree is in a particular neighborhood of the original tree. Specifies this is for phylogenetic trees but results can be used in general.

[7] Bryant, D., Filimon, F., and Gray, R. D. (2004). Untangling our past: Languages, trees, splits and networks. In R. Mace, C. Holden, S. Shennan. (eds) *The Evolution of Cultural Diversity: Phylogenetic Approaches*, UCL Press (in press). Creates a distance matrix that they can get splits from to create a tree of evolution of language. Definition of *splits graph*.

[8] Buneman, P. (1971). The recovery of trees from measures of dissimilarity. In F.R. Hodson, D.G. Kendall, P. T., editor, *Anglo-Romanian Conference on Mathematics in the Archaeological and Historical Sciences*, pages 387 – 395. University Press. Contains definition of *split* and *compatibility*. Defines a tree in terms of the splits then proves this representation is unique. Earliest reference (possibly source) of four-point condition.

[9] Buneman, P. (1974a). A characterization of rigid circuit graphs. *Discrete Mathematics*, 9:205 – 212. Buneman considers reconstruction of trees from raw data (e.g. looking at differences in genome sequence data positions). To this end he employs the graph-theoretic solution concept of intersection graphs, rigid circuit graphs, and cliques to come up with an algorithm to construct the tree from the raw data. Although he does provide a constructive algorithm, it is different from my own work in that it relies on the raw data and cannot easily be generalized to splits. The reason for this is due to the fact that the subsplits encode information about several leaves/species in one subsplit whereas the date just gives information on one species. To use Buneman's construction with subsplits would force nodes of the tree to be halves of the subsplits rather than individual taxa.

[10] Buneman, P. (1974b). A note on the metric properties of trees. *Journal of Combinatorial Theory*, 17:48 – 50. Looks like the first official reference to the four-point condition: a graph is a tree iff it satisfies the four-point condition. It also stipulates that we can construct a tree on any set that satisfies the four-point condition.

[11] Day, W. H. and McMorris, F. (2003). *Axiomatic Consensus Theory in Group Choice and Biomathematics*. SIAM, Philadelphia.

[12] Dobson, A. J. (1974). Unrooted trees for numerical taxonomy. *Journal of Applied Probability*, 11:32–42. A proof of four-point condition independent of Buneman. Argues for the usefulness of looking at unrooted trees instead of rooted trees for phylogenetic trees.

[Felsenstein] Felsenstein, J. PHYLIP. webpage http://evolution.genetics.washington.edu/phylip.html. Website where one can get the free program PHYLIP for inferring phylogenies.

[13] Felsenstein, J. (1983). Parsimony in systematics - biological and statistical issues. *Annual Review of Ecology and Systematics*, 14:313–333. Felsenstein discusses the issues in using the parsimony method for reconstructing evolutionary trees.

[14] Felsenstein, J. (2004). *Inferring Phylogenies*. Sinauer Associates, Inc., Massachusetts. Covers major methods for inferring phylogenies using numerical methods and statistical testing. Also discusses how to use phylogenies to make other inferences. Some discussion of rooting trees and four-point condition.

[15] Hall, B. (2001). *Phylogenetic Trees Made Easy: A How-To Manual for Molecular Biologists*. Sinauer Associates, Inc., Massachusetts. Mostly a manual to using computer programs to construct trees. Contained informational boxes discussing rooting trees, reliability of phylogenetic trees, tree searches, distance vs. character-based method, and models.

[16] Hammerstein, P. and Selten, R. (1994). Game theory and evolutionary biology. *Handbook of Game Theory with Economic Applications*, 2:929 – 993. uses two person games to model evolution.

[Hendy] Hendy, M. Hadamard conjugation: an analytic tool for phylogenetics. Slides for presentation. Definition of *Hadamard matrix* and statement of conjecture. Used to find tree with rate of mutation data with respect to parsimony methods.

[Huson] Huson, D. Splits tree 3.2. BiBiServ webpage http://bibiserv.techfak.uni-bielefeld.de/splits/. Website for the software that analyzes and visualizes distance data of phylogenetic trees.

[17] Kar, A. (2002). Axiomatization of the shapley value on minimum cost spanning tree games. *Games and Economic Behavior*, 38:265–277.

[18] Leclerc, B. and Makarenkov, V. (1998). On some relations between 2-trees and tree metrics. *Discrete Mathematics*, 192:223 – 249. Compared 2-trees with a tree function that can be thought of the weight of a (sub)tree.

[19] McFadden, C. (2005). Phylogenetic tree on soft coral genus *Alcyonium*. Prof. Catherine McFadden constructed this tree on soft coral. It is currently unpublished data that was given to me to use as an example of a phylogenetic tree in my thesis in April 2005.

[20] Nehring, K. and Puppe, C. (2002). A theory of diversity. *Econometrica*, 70(3):1155 – 1198. This article discusses how diversity could be measured. They use a multi-attribute approach which leads to representing the results in the form of a tree. Phylogenetic trees naturally fall into this category so they are discussed but there is no mention of the Shapley value or other game theoretic solution concepts.

[Pachter and Speyer] Pachter, L. and Speyer, D. E. Reconstructing Trees from Subtree Weights.

[21] Patrinos, A. and Hakimi, S. (1972). The distance matrix of a graph and its tree realization. *Quarterly of Applied Mathematics*, 30:255 – 269. Considers various distance matrices and classifies properties of matrix that construct trees or hypertrees. Generalizations of four-point condition.

[22] Semple, C. and Steel, M. (2002). A characterization for a set of partial partitions to define an $x$-tree. *Discrete Mathematics*, 247(1-3):169 – 184. Closely related to my thesis work, this paper considers reconstructing $X$-trees from partial partitions of the leaf set. An $X$-tree is a tree with leaves and nodes of degree 2 labeled and the partial partitions are just subsplits with multiple sections. They provide criteria for the existence of a unique $X$-tree that can be displayed by these partitions but does not give a constructive proof. Their proofs rely on graph theoretic ideas about the chords of cycles.

[23] Semple, C. and Steel, M. (2003). *Phylogenetics*. Oxford University Press, New York. Graduate level mathematics behind creating phylogenetic trees.

[24] Shapley, L. S. (1953). A value for n-person games. In *Ann. Math. Studies*, volume 28, pages 307–317. Princeton University Press, Princeton, N.J. This paper introduces the Shapley value which was the main concept used in my summer research paper included in the appendix.

[25] Weitzman, M. L. (1992). On diversity. *The Quarterly Journal of Economics*, 107(2):363–405. This article seems to be one of the first formally discussing the necessity for formal measures of diversity in many aspects of life. He uses the minimum distance between a taxon and a group to be the measure of diversity and discusses how this measure can construct a tree if it satisfies an analogy to the four-point condition. He also discusses reasonable axioms that any measure should satisfies (monotonicity, link property) plus some that are more specific (twin property, continuity in distances, etc.). He concludes his paper by showing how this could be helping in a biological setting such as trees.

[26] Weitzman, M. L. (1998). The noah's ark problem. *Econometrica*, 66(6):1279–1298. This paper focuses on applying "economic" theory to biodiversity. He constructs a model using expected diversity and utility functions to figure out optimal conservation strategies. He concludes that extreme measures (putting all your resources into a few species) are the most effective for maximizing the chance of maintaining biodiversity.