Decision Aiding Software Using FCA

Florent Domenach and Ali Tayari

Computer Science Department, University of Nicosia, 46 Makedonitissas Av.,
P.O.Box 24005, 1700 Nicosia, Cyprus, `domenach.f@unic.ac.cy`

Abstract. The consensus problem arises from social choice theory and systematic biology where we are looking for the common information shared by a series of trees. In this paper we present a decision aiding software to help systematic biologist to choose the consensus function the most appropriate for their need. This software is based on a previous study between consensus functions and axiomatic properties, and their underlined concept lattice.

1 Introduction

The consensus problem, which [11] deemed a "problem for the future", consists of summarizing a series of structures, usually trees, into one representative structure. Axiomatic studies of consensus functions is often [26] described as an "ideal situation [in which] the researcher formulates a list of desirable axioms that a consensus function should satisfy, and search for the best method that satisfies these axioms" [33]. We present here a software following this approach almost to the letter. Unfortunatly, it is still missing critical GUI features and is not available yet.

The motivation for the software is originating from the separation existing between theorizers and practitioners of consensus theory, what [7] denotes as abstract consensus theory and concrete consensus theory. On one hand, mathematicians are developing sophisticated mathematical tools. The modern development of the consensus problem originates from Arrow's work [3] (followed by [25]) who considered the problem of aggregating votes and showed that any voting system is either inconsistent, arbitrary or unstable. Since then, a lot of functions, together with a set of equivalent axioms, were developed (see [15, 22] for a comprehensive survey).

On the other hand, practitioners like systematic biologists are rarely using more than a handful of consensus functions. If you consider the most popular software available like PAUP*[1] [37] (majority), PHYLIP[2] [20] (majority, strict), or COMPONENT 2.0[3] [30] (strict, majority-rule, loose, Nelson and Adams consensus trees), only a handful are available for use. It was pointed out [38] that this gap between the two communities was detrimental to both.

---

[1] http://paup.csit.fsu.edu/

[2] http://evolution.genetics.washington.edu/phylip.html

[3] http://taxonomy.zoology.gla.ac.uk/rod/cpw.html

The goal of this paper is to present an approach – based on FCA – to the consensus problem that would fill the gap between both communities. We created a software that asks the user to think of desirable properties that a consensus method should possess, and then we advise on which consensus function satisfying these properties he/she should use. Each step is described in detail in the paper.

This paper is organized in four sections, the first one being this introduction. In Section 2, we give a precise definition of the consensus problem, as well as the definitions of the consensus functions (Section 2.1) and of the axiomatic properties (Section 2.2) that we implemented. We present in Section 3 the structure of our program, and explain for every step why and how we are doing it. Finally a brief conclusion is given in Section 4.

2 The Consensus Problem

Consider a finite set S, $|S| = n$. In phylogeny, the elements of S are called operational taxonomic units, or *taxa*. A hierarchy H on S, also called n-tree, is a family of subsets of S (called the *classes* or clusters of H) such that $S \in H$, $\emptyset \notin H$, $\{s\} \in H$ for all $s \in S$, and $A \cap B \in \{\emptyset, A, B\}$ for all $A, B \in H$. We will indifferently use the terms trees or hierarchies in the paper. We denote the set of all hierarchies on S by $\mathcal{H}$. Fig. 1 shows the graphical representation of different trees; usually the internal nodes are simply denoted by the leaves underneath.

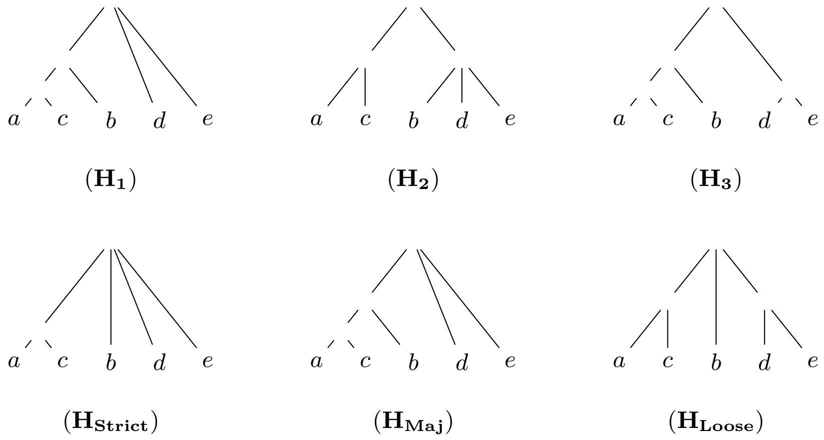
Consensus trees are summarizations of the information shared by two or more classification trees of the same set of taxa. Given a profile H^* of trees on S, i.e. a series of trees, we want to know what they have in common - we want to aggregate H^* in a unique tree H. We consider in this paper the case where all the trees of the profile are defined on the same set of taxa, as the generalization to super-trees [34] (where the trees can have different sets of taxa) can create computational problems.

2.1 Consensus Functions

Let $H^* = (H_1, H_2, ..., H_k)$ be a profile of hierarchies on S, and K will denote the set of indices of the hierarchies of H^*, $K = \{1, ..., k\}$. Formally, a *consensus function* on $\mathcal{H}$ is a map $c : \mathcal{H}^k \to \mathcal{H}$ with $k \geq 2$ and $\mathcal{H}^k$ the k cartesian product, which associate to any profile H^* a unique hierarchy consensus, $c(H^*)$. We do not aim to have an exhaustive list of consensus functions, a classification based on refinement is available in [13]. Consensus functions can be divided in three main categories:

Quota-based consensus functions. Consider a grouping and the associated index defined as:

$$N_{H^*}(A) = \{i \in K : A \in H_i\} \text{ and } n_{H^*}(A) = |N_{H^*}(A)|$$



$(\mathbf{H_1})$ $(\mathbf{H_2})$ $(\mathbf{H_3})$

$(\mathbf{H_{Strict}})$ $(\mathbf{H_{Maj}})$ $(\mathbf{H_{Loose}})$

Fig. 1. Different trees defined on the set of taxa $S = \{a, b, c, d, e\}$. For the profile $H^* = (H_1, H_2, H_3)$, the strict consensus tree is given by (H_{Strict}), the majority by (H_{Maj}) and the loose by (H_{Loose})

We associate the consensus function $c_{(p)} : \mathcal{H}^k \rightarrow \mathcal{H}$ to the index n_{H^*} for any $p \in K$. A subset A is called *p-frequent* if $n_{H^*}(A) \geq p$, and the *p-frequent consensus* of H^*, denoted as $c_{(p)}(H^*)$, is the family of all p-frequent subsets. Quota-based consensus functions are particular cases of federation consensus functions [23]. Recall that a federation (simple game) is a family $\mathcal{F}$ of subsets of K such that $A \in \mathcal{F}, B \supseteq A$ imply $B \in \mathcal{F}$. A *federation consensus function* $c_\mathcal{F}$ is then defined as $c_\mathcal{F}(H^*) = \vee_{S \in \mathcal{F}}(\cap_{i \in S} H_i)$. If we take the simple case where, for some $j \in K$, $\mathcal{F} = \{S \subseteq K : j \in S\}$, we have $c_\mathcal{F}(H^*) = H_j$, a single hierarchy dictating the result of the consensus, the so called *projection* consensus function.

$$\text{Projection: } \exists j \in K : Prj(H^*) = H_j$$

When we extend this to a subset J of K, we have the *oligarchic* consensus function using $\mathcal{F} = \{S \subseteq K : J \subseteq S\}$, and $c_\mathcal{F}(H^*) = \bigcap_{j \in J} H_j$.

$$\text{Oligarchy: } \exists J \subseteq K : Ol(H^*) = \bigcap_{j \in J} H_j$$

In the family of quota-based consensus functions, one can notice $c_{(k)}(H^*) = \bigcap_{i \in K} H_i$ the set of classes present in all trees of the profile, i.e. the *strict* consensus function [36]. In Fig. 1, (H_{Strict}) is the strict consensus of the profile (H_1, H_2, H_3).

$$\text{Strict: } Str(H^*) = \bigcap_{i \in K} H_i$$

If we take $p = \lceil \frac{k+1}{2} \rceil$, the smallest natural number greater than $\frac{k}{2}$, we have the *majority* consensus function [24] which considers clusters appearing in at least half of the trees. An example of the majority consensus function is given in Fig. 1, where $(H_{Maj}) = Maj(H_1, H_2, H_3)$.

$$\text{Majority: } Maj(H^*) = \{X \subseteq S : n_{H^*}(X) > \frac{k}{2}\}$$

Unfortunately, if p is less than $\lceil \frac{k+1}{2} \rceil$, it cannot be guaranteed that the resulting family will be a tree. In order to keep the structure of a tree, different strategies can be used.

Frequency-based consensus functions A first approach considers the idea of compatibility, i.e. two sets A and B are *compatible* if $A \cap B \in \{\emptyset, A, B\}$, denoted as $A \| B$, and a set A is compatible with a hierarchy H if it is compatible with every cluster of H (or, equivalently, if $A \cup H \in \mathcal{H}$). We then can define a consensus function called *loose* consensus [6] (originally called combinable component [12], also called semi-strict) which considers subsets as long as they are compatible with every tree of the profile. Fig. 1 shows (H_{Loose}), the loose consensus tree obtained from (H_1, H_2, H_3).

$$\text{Loose: } L(H^*) = \bigcup \{X \subseteq S : \exists j \in K, X \in H_j \text{ and } \forall i \in K, X \cup H_i \in \mathcal{H}\}$$

The loose consensus function was extended by [18] to two different consensus functions. The first one is combining the classes obtained by the majority consensus function with those of the loose consensus function:

$$\text{Loose and Majority Function Property: } LM(H^*) = Maj(H^*) \cup L(H^*)$$

The second extension is to add classes that are more often compatible than not. Define $N_{H^*}(\overline{X}) = \{i \in K : X \cup H_i \notin \mathcal{H}\}$ as the set of trees not compatible with a subset X, then the *majority (+)* consensus function will take subsets that are more often compatible than incompatible. It obviously contains all the classes obtained by the majority function and by the loose function.

$$\text{Majority-rule (+) : } Maj^+(H^*) = \{X \subseteq S : |N_{H^*}(X)| > |N_{H^*}(\overline{X})|\}$$

Consider the weight function $w(X) = n_{H^*}(X) - 1$ on classes. The *Nelson-Page* consensus tree is the tree constructed from the clique G containing the components most frequently replicated in the profile. If two or more cliques have the same, maximal number of replications of components, then the consensus tree is constructed from those components common to all those cliques. In the literature, the Nelson-Page tree [27, 29] has often been confused with the strict consensus tree.

The *frequency difference* consensus function consider the subsets of S that are more frequent than any other subsets non-compatible.

$$\text{Freq. Diff.: } FD(H^*) = \{X : n_{H^*}(X) > max_{Y \text{ not compatible with } X}\{n_{H^*}(Y)\}\}$$

Previous consensus functions may miss some structural features of the trees, particularly if the data is noisy. For example, a desirable feature would be that if two taxa are closer than a third one, we want these two taxa to be separated from the third one in the consensus hierarchy - which is what Adams' function [1] achieves. Historically the first one, an *Adams* consensus tree contains the nestings common to all trees in a profile. X nests in Y in H, denoted as $X <_H Y$ if and only if $X \subset Y$ and there is $Z \in H$ such that $X \subseteq Z$ and $Y \nsubseteq Z$. $\pi(H)$ is the maximal cluster partition for H with blocks equal to the maximal clusters of H. Adams' consensus function is best described algorithmically (from [13]):

```
Procedure AdamsTree($H_1, ..., H_k$)
    Construct $\pi(H)$, the product of $\pi(H_1), ..., \pi(H_k)$.
    For each block $B$ of $\pi(H)$ do
        AdamsTree($H_1|_B, ..., H_k|_B$)
```

Distance-based consensus functions Another consensus family is based on distance, either as a height function, or as distance between trees. *Durchschnitt* [28] consensus function takes the intersection of all classes at the same height. The canonical height $\eta_0(X)$ of a class $X \subseteq S$ is defined as $\eta_0(S) = 0$ and $\eta_0(X) = h$ if and only if there is a maximal sequence $S \supset X_1 \supset ... \supset X_{h-1} \supset X_h = X$. Define $\omega = min_{i \in K} max_{X \in H_i} \eta_0(X)$ as the height of the smallest tree of the profile.

$$\text{Durchschnitt: } Dur(H^*) = \bigcup_{j=1}^{\omega} \{ \bigcap_{i \in K} X_i : X_i \in H_i \text{ and } \eta_0(X_i) = j \}$$

The median and asymmetric median consensus functions both use a distance between trees, i.e. a distance on $\mathcal{H}$. The *median* consensus is the tree minimizing the distance of the symmetric difference from it to every tree of the profile. The median consensus was extensively studied, particularly in the case of semi-lattices [35] (as trees can be seen as semi-lattices).

$$\text{Median: } Med(H^*) = min_{H \in \mathcal{H}} \sum_{i=1}^{k} |H \triangle H_i|$$

The *asymmetric median* consensus [32] on the other hand is the tree minimizing the distance between each tree and the consensus tree, i.e. minimizing the number of classes in H_i that are not present in $c(H^*)$.

$$\text{Asymmetric Median: } AMed(H^*) = min_{H \in \mathcal{H}} \sum_{i=1}^{k} |H_i - H|$$

2.2 Axiomatic Properties of Consensus Functions

Historically, consensus functions were studied through a series of (desirable) axioms proved to be equivalent to the function. Arrow's pioneer work proved the impossibility of a non-dictatorial consensus function satisfying fundamental axioms (transitivity, Pareto and independence of irrelevant alternatives) on linear orders. We implemented a series of axioms that a user may find desirable or undesirable.

A consensus function is Pareto relatively to a specific kind of relationships (classes, triplets, nestings) when the consensus tree will contain the relationship present in all the trees, i.e. will contain the intersection of the trees of the profile with respect to the relationship. For example, when we are interested in the common classes, we have the *Pareto optimal* [31] axiom:

$$\text{Pareto Optimality: } (\forall X \subseteq S)(X \in \bigcap_{i=1}^{k} H_i \Rightarrow X \in c(H^*))$$

Trees can also be defined [14] through triplets $ab|c$, $a, b, c \in S$, denoting the grouping of a and b relative to c. We say that $ab|c \in H$ if there exists a class $X \in H$ such that $a, b \in X$ but $c \notin X$. The Pareto property on triplets is that a common separation of two taxa from a third taxon among every input tree must be respected and applied in the consensus tree.

$$\text{Ternary Pareto Optimality: } (\forall x, y, z \in S)((\forall i \in K)(xy|z \in H_i) \Rightarrow xy|z \in c(H^*))$$

Adams [2] extended that idea to nestings, where if two clusters are separated from each other in every input tree, therefore they must also be separated in the consensus tree:

$$\text{Nesting Preservation: } (\forall \emptyset \neq X, Y \subseteq S)((\forall i \in K)(X <_{H_i} Y) \Rightarrow (X <_{c(H^*)} Y))$$

Conversely, a consensus function is *co-Pareto* for a particular relationship if one can find every relationship of that kind of the consensus tree in one or more tree of the profile. Every cluster from the consensus tree must appear in at least one of the input tree, or in other words it should be a member of the union of all input trees. We will consider here only co-Pareto optimally for classes.

$$\text{co-Pareto Optimality: } (c(H^*) \subseteq \bigcup_{i=1}^{k} H_i)$$

In order to characterize his consensus function, Adams introduced a reciprocal property of nesting preservation, although stronger than just a co-Pareto property. It states that if two subsets are nested in the consensus tree, they must be nested in all the trees of the profile.

$$\text{Strong Presence: } (\forall \emptyset \neq X, Y \subseteq S)(X <_{c(H^*)} Y \Rightarrow (\forall i \in K)(X <_{H_i} Y))$$

It happened that Strong Presence property was too constraining, so instead of considering all possible nested subsets, Adams considered only the nested classes. Any two clusters of the consensus tree that are separated from each other must also be separated in every input tree.

$$\text{Qualified Strong Presence: } (\forall X, Y \in c(H^*))(X <_{c(H^*)} Y \Rightarrow (\forall i \in K)(X <_{H_i} Y))$$

Qualified strong presence was weakened to consider the clusters of the consensus tree to be nested in S in each tree of the profile:

$$\text{Upper Strong Presence: } (\forall X \in c(H^*))(X <_{c(H^*)} S \Rightarrow (\forall i \in K)(X <_{H_i} S))$$

The *dictatorship* property (an input tree dictates over the consensus tree by having all of its clusters included in the consensus tree) is often consider undesirable; however, this can change if there is a particular tree that can be consider an oracle, i.e. for which we want the consensus tree to refine it.

$$\text{Dictatorship: } (\exists j \in K)(\forall X \subseteq S)(X \in H_j \Rightarrow X \in c(H^*))$$

Another desirable property, also called faithful, is the following: for every group of clusters containing only one cluster from each input tree there must be a cluster in the consensus tree such that it includes the intersection of the group of the group of clusters and it is included in the union of the groups of the group of clusters.

$$\text{Betweenness: } (\forall i \in K \text{ with } X_i \in H_i)(\exists Y \in c(H^*))(\bigcap_{i=1}^{k} X_i \subseteq Y \subseteq \bigcup_{i=1}^{k} X_i)$$

3 Decision Aiding Software

We used Formal Concept Analysis (FCA) [21] as our formal background. FCA is particularly suitable as it provides a structure on the power set of attributes, here the consensus functions and axioms, and allow calculations of distances on that structure. Since we assume the reader familiar with FCA, we will only briefly recall main terminologies and results used in our program: a *formal context* (G, M, I) is defined as a set G of objects, a set M of attributes, and a binary relation $I \subseteq G \times M$. $(g, m) \in I$ is read as "object g has attribute m". To this formal context, one can associate to a set of objects $A \subseteq G$ its intension $A' = \{m \in M : \forall g \in A, (g, m) \in I\}$ of all properties shared by A. Dually, we can define $B' = \{g \in G : \forall m \in B, (g, m) \in I\}$, the extension of a set of properties $B \subseteq M$. A pair $(A, B), A \subseteq G, B \subseteq M$, is a *formal concept* if $A' = B$ and $B' = A$. The set of all formal concepts, ordered by inclusion, forms a lattice [5], called *concept lattice*. For more terms and definitions on lattice theory, one can refer to [10, 16].

This D.A. software has three different functional layout (see Fig. 2): a pre-processing is first done on consensus functions and axioms in order to create the context that then will be used, with the associated lattice, in order to advise users on which consensus function to use. The last layer is concerned with the obtainment of the tree itself from some input profile.

3.1 Pre-processing

The first layer of the D.A. software concerns the pre-processing of the data that will be used. In order to insure scientific validity of the decision aiding, we implemented the previous consensus functions of Sec. 2.1 and the axiomatic properties

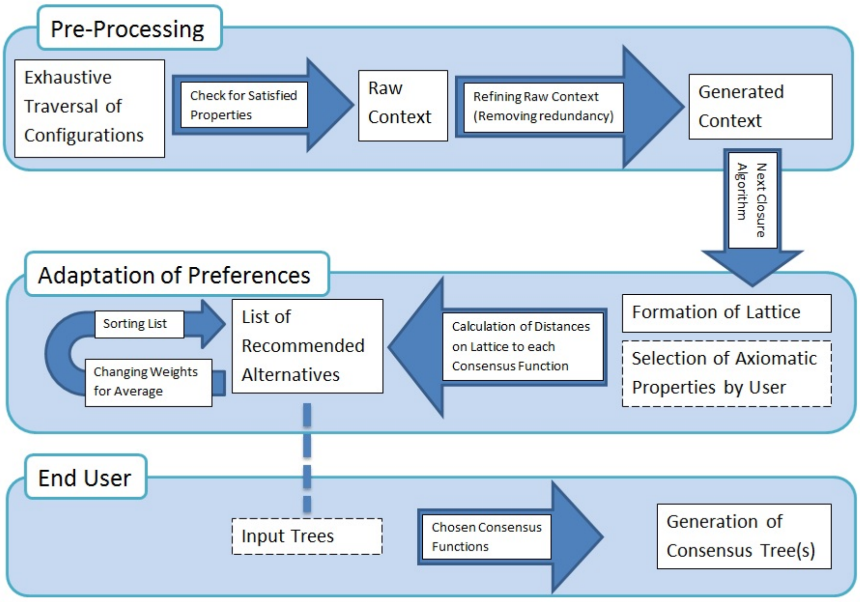


Fig. 2. D.A. software functional layout.

of Sec. 2.2 in C++ on a laptop Intel Core i5, 2.3 GHz. Initially, it generates all possible hierarchies based on a given set of n taxa, and traverses through all possible profiles of k hierarchies, together with all possible consensus trees. Then we exhaustively list what we called configurations, each configuration is a pair consisting of a profile and a consensus tree. Every configuration was systematically compared against axiomatic properties and consensus functions in order to create a first (raw) context. Attributes of the context are the consensus functions and the axiomatic properties, while the objects are every possible configuration. We discussed in [17] the implications generated by the context.

During the pre-processing phase, we encountered a series of computational challenges, as the number of n-trees grows exponentially [19] and some consensus functions are NP-hard [32]. We were able to exhaustively investigate the configurations only up to $n = 5$, for which we obtained around 9.57×10^{12} configurations. Since the running time of the simulation increases exponentially with slight addition to n or k, in order to have partial results, controlled randomly selected configurations were chosen in order to have a more accurate - and so a more refine - context.

3.2 Underlined Structure

Given the number of objects in our context (over one trillion), we first eliminate duplicates. If several configurations share the same attributes, we simply keep the first one as representative. No information is lost as we are interested in the structure of the attributes, and the objects (the configurations) sole purpose is to systematically investigate this structure. Our simplified context has 5379 objects for 23 attributes, and Fig. 3 shows the overall concept lattice, having 3718 concepts. In order to derive the lattice, we followed The Next Closure [21] algorithm. This algorithm uses the lectic order on the set of attributes M, which is a total order on $\mathcal{P}(M)$. Given two subsets A and B of M, A is said to be lectically smaller than B at position i, and we denote it by $A \prec_i B$, if and only if $i = min(A \Delta B)$ and $i \in B$. Finally, we say that A is lectically smaller than B if $A = B$ or $A \prec_i B$ for some $i \in K$. We used Next Closure algorithm as it is an efficient and easy to implement.

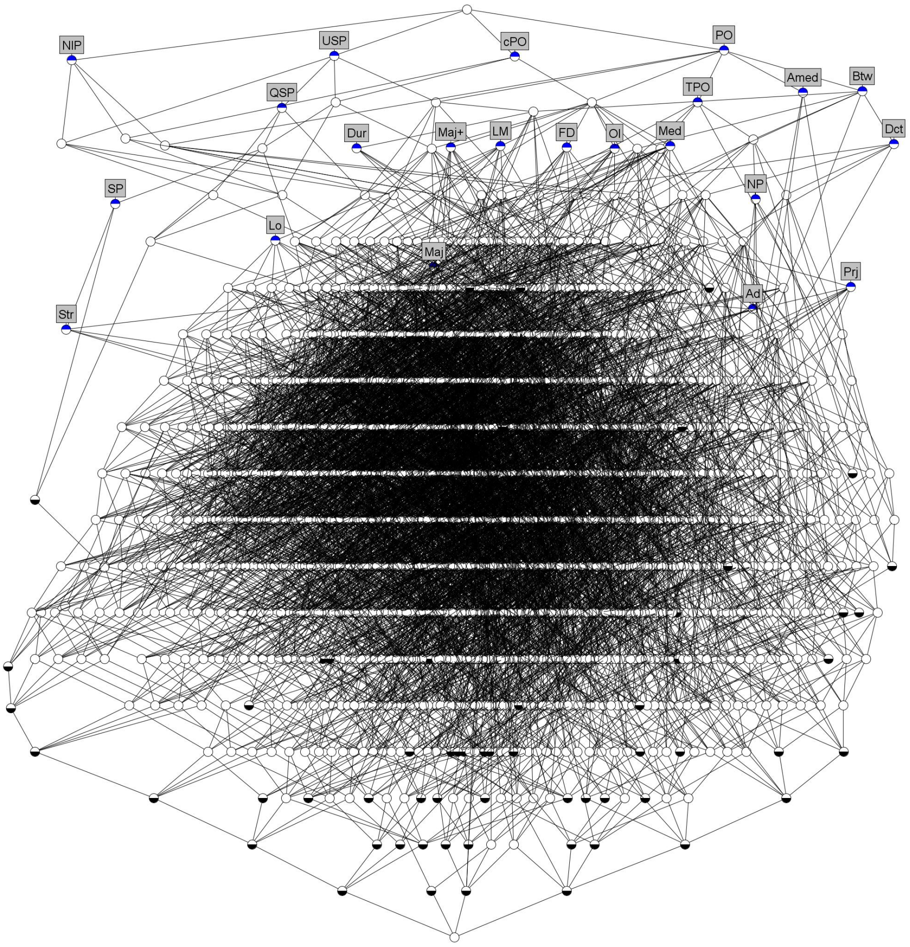


Fig. 3. Concept lattice associated with the configurations with minimal labeling of the properties (drawn with ConExp [39]).

After constructing the list of concepts and listing them in ascending order, the program also keeps track of the children and parents of each concept of the lattice. The user can then select a set of axiomatic properties depending on the one he/she finds desirable or undesirable: each axiom can be preferred (positive), disliked (negative), or neutral. Based on that input, the program finds the meet of the selected properties, i.e. the concept C associated to his/her choices. Concept C is the smallest concept containing all the positive user's choices and no negative ones if it exists. If the user's choices are conflicting, i.e. C doesn't exist, positive choices will be given priority over negative ones.

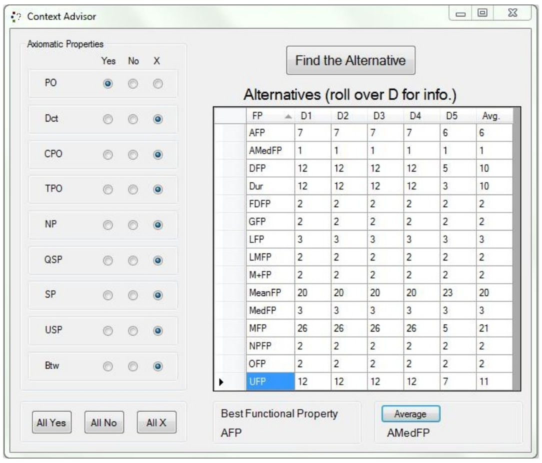
3.3 Distances in the lattice

In order to advise which consensus function would be suitable depending on the user's choices, for each consensus function, we first find the smallest concepts C_i containing that consensus function. Then we used different distances between C (the concept representing the user's choice) and each C_i (the concepts associated with consensus function i) in order to find the consensus function the closest to the user's choice. The use of different distances lets the user freely choose which distance is more suitable.

We can consider two types of distances in a lattice: distances based on concepts and distances on the covering graph (or Hasse diagram) of the lattice. For the first type, we used the *distance of the symmetric difference* between concepts, $d_1(C, C_i) = |C \Delta C_i|$, i.e the number of properties present in either C or C_i but not in both. For the second type, distances in the covering graph, we considered four distinct distances:

- *Any Path Distance*: weight of the shortest path (topological distance) between the corresponding attribute concepts; the closer the two concepts are in the graph, the greater their likelihood.
- *Any Path Distance Without* $\bot_L$ *and* $\top_L$: we remove the top and the bottom concepts of the lattice to compute the topological distance because such concepts don't bear any information (even if 1_L can have attributes associated, it still doesn't have any meaning). It is particularly important when we consider the co-atoms of the lattice (such as Pareto Optimal or co-Pareto Optimal, see Fig. 3), as the shortest path could go through 1_L and short-circuit the "real" distance.
- *Meet Distance*: It is the topological distance between C and C_i passing through their meet, i.e. the distance from C to $C \wedge C_i$ plus the distance from C_i to $C \wedge C_i$.
- *Join Distance*: Dual to the meet distance, it is the topological distance between C and C_i passing through their join.

Since each previous distance has its own advantages and disadvantages, we also implemented a weighted average distance for which the user can freely assign the weights. It is a weighted average of all the above distances based on user's preference. Fig. 4 shows an example of user's choice and the advised consensus function.



Axiomatic Properties

	Yes	No	X
PO	●	○	○
Dct	○	○	●
CPO	○	○	●
TPO	○	○	●
NP	○	○	●
QSP	○	○	●
SP	○	○	●
USP	○	○	●
Btw	○	○	●

Alternatives (roll over D for info.)

FP	D1	D2	D3	D4	D5	Avg.
AFP	7	7	7	7	6	6
AMedFP	1	1	1	1	1	1
DFP	12	12	12	12	5	10
Dur	12	12	12	12	3	10
FDFP	2	2	2	2	2	2
GFP	2	2	2	2	2	2
LFP	3	3	3	3	3	3
LMFP	2	2	2	2	2	2
M+FP	2	2	2	2	2	2
MeanFP	20	20	20	20	23	20
MedFP	3	3	3	3	3	3
MFP	26	26	26	26	5	21
NPFP	2	2	2	2	2	2
OFP	2	2	2	2	2	2
UFP	12	12	12	12	7	11

Best Functional Property
AFP

AMedFP

Fig. 4. Screen shot of the second layer of the software, with an example of user's choices.

3.4 Decision Aiding

In the third layer, the D.A. Software recognizes input trees which are given in Newick format. The Newick tree format is a well-known representation of graph-theoretical trees which denotes trees using parentheses and commas. The simplicity and standard nature of Newick makes it a suitable method for scientists to provide the software with their input. There are several ways through which trees can be represented, however the representation that contains only the information about the leaves are recognized as the valid ones. For example, in Fig. 1, (H_1) has the Newick format $(((a, c), b), d, e)$, while (H_2) is $((a, c), (b, d, e))$.

Upon selection of consensus functions by the user, the D.A. software generates the unique (or set of all possible consensus trees) for the selected functions, so that the user can compare them with each other. Using this feature, the user is able to find out which model would be more suitable for the nature of their work, for which he/she will be provided with respective consensus tree(s). This allows the user to have a narrowed list of candidates for the representative consensus trees as well as having a hands-on experience to find out the most suitable functional property and consensus tree.

4 Conclusion and Future Work

In this paper, we presented a decision aiding software which explore via Formal Concept Analysis the space of consensus functions and their axioms. It provides the user with means to generate consensus tree(s) representative(s) depending on their choices. It initially imports the raw context obtained via pre-processing, constructs the associated lattice and, depending on the user's preferences, advise based on distances in the lattice on which function to use. Upon selection of functions, the program generates the consensus trees of the collection of user's input tree using selected functional properties.

In continuation of this project, we are planning to expand the capabilities of this software. Firstly, besides the (rooted) trees that are currently supported as input and output, the program will be able to support super-trees as well as unrooted trees as its input and output. Another possibility would be the addition of other types of structures of sets such as pyramids [9], weak-trees [4], and, more generally, lattices. In addition, the concept of *independence* and *neutrality* as axiomatic properties are planned to be incorporated. Moreover, other commonly used consensus functions are going to be added to the result, therefore with a further refined and exhaustive approach, the program's precision and usefulness would be improved.

References

[1] Adams III, E.N.: Consensus Techniques and the Comparison of Taxonomic Trees. Systematic Zoology **21** (1972) 390–397

[2] Adams III, E.N.: N-trees as nestings: Complexity, similarity, and consensus. J. Classif **3** (1986) 299-317

[3] Arrow, K.J.: Social Choice and Individual Values. Wiley, New York (1951)

[4] Bandelt, H.-J., Dress, A.: Weak hierarchies associated with similarity measures: an additive clustering technique Bull. Math. Biology **51** (1989) 133-166

[5] Barbut, M., Monjardet, B.: Ordres et classification: Algèbre et combinatoire (tome II). Hachette, Paris (1970)

[6] Barthélemy, J.-P., McMorris, F.R., Powers, R.C: Dictatorial consensus functions on n-trees. Math. Soc. Sci. **25** (1992) 59-64

[7] Barthélemy, J.-P., Brucker, F.: Average Consensus in Numerical Taxonomy. In Data analysis, Eds. W. Gaul, O. Opitz, and M. Schader, Springer (2000) 95-104

[8] Bertrand, P.: Set Systems and Dissimilarities. Europ. J. Combinatorics **21** (2000) 727-743

[9] Bertrand, P., Diday, E.: A visual representation of compatibility between an order and a dissimilarity index: the pyramids. Comput. Stat. Quart. **2** (1985) 31-44

[10] Birkhoff, G.: Lattice Theory, 3rd ed. Amer. Math. Soc., Providence (1967)

[11] Bock, H.H.: Classification and clustering: Problems for the future. In Diday, E., Lechevallier, Y., Schader, M., Bertrand, P., Burtschy, B.: New approaches in classification and data analysis. Springer-Verlag, Berlin (1994) 3–24

[12] Bremer, K.: Combinable component consensus. Cladistics **6** (1990) 369-372

[13] Bryant, D.: A Classification of Consensus Methods for Phylogenetics. In: Janowitz, M., Lapointe, F.J., McMorris, F., Mirkin, B., Roberts, F. (eds.) Bioconsensus, DIMACS (2003) 163-184

[14] Colonius, H., Schulze, H.-H.: Tree structure for proximity Data. Brit. J. of Math. Stat. Psych. **34** (1981) 167-180

[15] Day, W.H.E., McMorris, F.R.: Axiomatic Consensus Theory in Group Choice and Biomathematics. Siam, Philadelphia (2003)

[16] Davey, B.A., Priestley, H. A.: Introduction to Lattices and Order, 2nd ed. Cambridge University Press (2002)

[17] Domenach, F., Tayari, A.: Implications of Axiomatic Consensus Properties. To appear (2012)

[18] Dong, J., Fernández-Baca, D., McMorris, F.R., Powers. R.C.: An Axiomatic Study of Majority-rule (+) and associated Consensus Functions on Hierarchies. Disc. App. Math. **159** (2011) 2038-2044

[19] Felsenstein, J.: The Number of Evolutionary Trees. Syst. Zool. **27** (1978) 27-33

[20] Felsenstein, J.: PHYLIP - Phylogeny Inference Package (Version 3.2). Cladistics **5** (1989) 164-166

[21] Ganter, B., Wille, R.: Formal Concept Analysis : Mathematical Foundations. Springer (1996)

[22] Hudry, O., Monjardet, B.: Consensus Theories. An Oriented Survey. Math. Sci. hum **190** (2010) 139-167

[23] Leclerc, B. Monjardet, B.: Latticial theory of consensus. In: Barnett, V., Moulin, H., Salles M., Schofield N. (eds.) Social choice, Welfare and Ethics. Cambridge University Press, Cambridge (1995) 145-159

[24] Margush, T., McMorris, F.R.: Consensus n-trees. Bull. Math. Biol. **43** (1981) 239-244

[25] May, K.O.: A Set of Independent Necessary and Sufficient Conditions for Simple Majority Decision. Econometrica **20** (1952) 680-684

[26] McMorris, F.R.: : Axioms for consensus functions defined on undirected phylogenetic trees. Math. Biosciences **74** (1985) 77-80

[27] Nelson, G.: Cladistic analysis and synthesis: Principles and definitions, with a historical note on Adanson's Famille des Plantes (1763-1764). Syst. Zool. **28** (1979) 1-21

[28] Neumann, D.A.: Faithful consensus methods for n-trees. Math. Biosci **63** (1983) 271-287

[29] Page, R.D.M.: Tracks and Trees in the Antipodes: A Reply to Humphries and Seberg. Syst. Zool. **39** (1990) 288-299

[30] Page, R.D.M.: User's manual for COMPONENT, Version 2.0. Natural History Museum, London (1993)

[31] Pareto, V.: Cours d'économie politique. F. Rouge, Lausanne (1896)

[32] Phillips, C., Warnow, T.J.: The aymmetric median tree - A new model for building consensus trees. Disc. App. Math. **71** (1996) 311-335

[33] Powers, R.C., White, J.M.: Wilson's theorem for consensus functions on hierarchies. Disc. Appl. Math. **156** (2008) 1321–1329

[34] Semple, M., Steel, C.: A supertree method for rooted trees. Disc. App. Math. **105** (2000) 147-158

[35] Sholander M.: Medians, Lattices, and Trees. Proceedings of the American Mathematical Society **5** (1954) 808-812

[36] Sokal, R.R., Rohlf, F.J., Taxonomic congruence in the Leptopodomorpha reexamined. Syst. Zool. **30** (1981) 309-325

[37] Swofford, D.L.: PAUP: Phylogenetic Analysis Using Parsimony, version 3.0. Illinois Natural History Survery, Champaign (1990)

[38] Wilkinson, M., Thorley, J.L., Pisani, D.E., Lapointe, F.-J., McInerney, James O.: Some Desiderata for Liberal Supertrees. In: Phylogenetic Supertrees: Combining Information to Reveal the Tree of Life, Kluwer Academic Publishers (2004) 564-582

[39] Yevtushenko, S.A.: System of data analysis "Concept Explorer". Proceedings of the 7th national conference on Artificial Intelligence KII-2000, Russia (2000) 127-134