A distance measure between plant architecture

Pascal Ferraro, Christophe Godin

A distance measure between plant architectures

Pascal Ferraro* and Christophe Godin

Plant Modelling Program – CIRAD, Programme de modélisation des plantes, TA/40E - 34398 Montpellier Cedex 5, France, (tel. 0467593853, Fax 0467593858)
ferraro@cirad.fr

Abstract: In many biological fields (e.g. horticulture, forestry, botany), a need exists to quantify different types of variability within a set of plants. In this paper, we propose a method to compare plant individuals based on a detailed comparison of their architectures. The core of the method relies on an adaptation of an algorithm for comparing rooted tree graphs, recently proposed by Zhang in theoretical computer science. Using this algorithm a distance between two plants is defined as the cost of transforming one into the other (using basic "edit operations"). We illustrate this method in three application fields and then compare it with other methods for quantifying plant similarity.

Key Word: **topological structure of plants / plant comparison / analytical method**

Définition d'une distance entre architectures de plantes

Résumé: Dans de nombreux domaines de la biologie (arboriculture, sylviculture, botanique), il est nécessaire d'étudier différents types de variabilité au sein d'une population de plantes. Nous proposons, dans ce papier, une méthode de comparaison des plantes basé sur une comparaison détaillée de leur architecture. Cette méthode est une adaptation d'un algorithme de comparaison d'arborescences, proposé récemment par Zhang en informatique théorique. Cet algorithme nous permet de définir une distance entre deux plantes comme le coût de la transformation de l'un en l'autre (à l'aide d'opérations élémentaires d'édition). Cette méthode est illustrée dans trois domaines d'application et elle est comparée à d'autres méthodes de quantification de la ressemblance entre plantes.

Mot Clé: **structure topologique des plantes / comparaison des plantes/ méthode analytique**

1. Introduction

The increasingly important role played by plant architecture in the structure/function modeling of plants generates a need for new investigational tools. Generic tools have already been developed to visualize plant architecture in 3-dimensions ([4,30]), to model the growth of plant architecture, e.g. ([20,25,5,21]), to measure plant architecture ([9,14,33]), and to explore and to analyze the plant [10]. This paper introduces a new tool for the comparison of plant architectures.

To compare two plants, a first approach consists of summarizing each individual by a small number of synthetic and global variables (e.g. fruit production, crown size, etc.). The similarity of two individuals is then reduced to the similarity between these synthetic variables. In forestry for instance, wood production and quality are usually assessed by measuring variables such as stem diameter, crown volume, branching density, etc. Comparing different wood qualities thus amounts to comparing these global variables. This defines *global comparison methods* in which the topological organization of plant entities is not taken into account.

On the other hand, domains exist in which plant topological structure plays an important role. In forestry for example, refining wood quality criteria leads foresters to consider more detailed descriptions of tree crowns, taking for instance into account the spatial distribution of branches along the stems or branch geometry (e.g. [19]). Similarly, in horticulture, determining the fruiting position in the tree crown leads to a better understanding of the fruiting habits and production parameters (e.g. [3]). In such cases, the notion of distance between individuals would naturally take into account the topological and spatial organization of plant entities. This defines *analytical comparison methods* which are based on a piece-by-piece comparison of plants [26].

In most applications, descriptions of plant architecture usually rely on a tree graph representation of topological structures [8]. An algorithm with bounded complexity has recently been proposed in theoretical computer science to compute a distance between tree graphs [43,44]. This distance is defined as the minimum cost of the sequence of elementary edit operations needed to transform one tree graph into the other. This paper proposes an analytical comparison method based on an adaptation of this algorithm to deal with plant architectures. The different methods used to tune the parameters of this algorithm are then reviewed and discussed. Finally, the use of this comparison algorithm is illustrated in three different application contexts.

2. Formal representation of plants as tree graphs

A plant can be considered as a set of botanical entities (e.g. internodes and nodes, growth units, annual shoots) the topological organization of which can be represented by a graph [8] (Figure 1). A graph $G=\{V,E\}$ consists of a set V of vertices and a set of edges E, each edge being represented by an ordered pair of vertices [29]. If (v_1, v_2) denotes an edge in E, the vertex v_1 is called the *father* of v_2 and the vertex v_2 is called the *son* of v_1 [29]. Vertices represent botanical

entities and edges correspond to the physical connections between these entities. Each vertex can be associated with one or several attributes that represent biological characteristics of the entity and consists of either a real number (e.g. entity diameter, length), or a symbol (e.g. entity type). Let α be a labeling function which associates a label from a finite or infinite set $\Sigma=\{a,b,c,...\}$ with each vertex. A distance d, called elementary distance, is supposed to be defined on labels. A distance on vertices of a graph can be defined using the distance on labels: $d(v_1,v_2)=d(\alpha(v_1), \alpha(v_2))$. Let λ be a unique symbol not in Σ, d is extended by defining quantities $d(\alpha(v_1),\lambda)$ and $d(\lambda,\alpha(v_2))$ so that d is a distance on $\Sigma \cup \{\lambda\}$. The distance $d(\alpha(v_1), \lambda)$ between the label of a vertex v_1 and the label λ is denoted by $d(v_1,\lambda)$ by convention.

In a plant, since each entity is physically attached to at most one parent entity, the topological structure is represented as a rooted tree graph, i.e. a graph in which every vertex except one, called the root, has only one father vertex. The root has no father vertex. In order to identify the different axes on a given plant, two types of edges between entities are distinguished: an entity can either precede (symbol '<') or bear (symbol '+') another entity. This form of plant description can now be used to present an analytical method for comparing plants.

3. Plant comparison method

A considerable amount of work has been performed on comparison algorithms for problems that can be modeled as data sequences [35]: in molecular biology [16,34], in speech or text recognition [23] or in code error correction [39], in plant modeling [11]. In the early seventies, Wagner and Fisher [42] presented an algorithm which computes the distance between two strings of characters as the minimum cost sequence of elementary operations needed to transform one string into the other. In order to define a distance between rooted tree graphs, Tai [37], Selkow [32] and Lu [24] proposed a generalization of the Wagner and Fisher algorithm with application in different fields [27,28,32]. All the tree graphs discussed in these papers are *ordered*, meaning that the sets of sons of any vertex are ordered sets. These algorithms cannot be applied directly to the problem of plant comparison since tree graphs used to represent plant topology are unordered [8]. However, recently, Zhang [43,44] proposed an algorithm in theoretical computer science for computing a distance between unordered rooted tree graphs based on Lu's method, by introducing a new hypothesis in the tree-graph transform. This paper briefly describes the main principle of the algorithm and illustrates several applications to plant comparison.

A distance measure between two trees T_1 and T_2 is defined by considering the minimum cost of elementary operations needed to transform T_1 into T_2. Three kinds of elementary operations, called *edit operations* [42] are considered: changing one vertex into another (note that this may change labels), deleting (i.e. making the sons of a vertex v become the sons of the father of v and then removing v from T_1) or inserting one vertex (i.e. the symmetric operation on T_2) (Figure 2*a*). In order to transform one tree graph into the other, all the vertices of T_1 and T_2 must be affected by at least one edit operation.

A cost function, called *local distance*, is defined for each edit operations s. The local distance assigns a non-negative real number $\gamma(s)$ to s:

- $\gamma(s)=d(v_1,v_2)$ if s changes the vertex v_1 into the vertex v_2,
- $\gamma(s)=d_{del}(v_1)=d(v_1,\lambda)$ if s deletes the vertex v_1 and,
- $\gamma(s)=d_{ins}(v_2)=d(\lambda,v_2)$ if s inserts the vertex v_2.

Since d is a distance, the following property is always satisfied:

$$d(v_1,v_2) \leq d_{del}(v_1) + d_{ins}(v_2) \quad (1)$$

Let S be a sequence of n edit operations $(s_1, s_2, \ldots, s_n)$ which transform one tree graph T_1 into another one T_2. The cost $\gamma(S)$ of a sequence of edit operations is defined by summing up the cost of the edit operations that compose S: $\tilde{a}(S) = \sum_{s_i \in S} \tilde{a}(s_i)$. The set of possible edit operation sequences which transform T_1 into T_2 is denoted by $\mathbf{S}$. The dissimilarity measure[1] $D(T_1,T_2)$ from a tree graph T_1 to a tree graph T_2 is then measured as the minimum cost of a sequences in $\mathbf{S}$:

$$D(T_1,T_2) = \min_{S \in \mathbf{S}} \{\tilde{a}(S)\}$$

In order to characterize the effect of a sequence of edit operations on a tree graph, Taï [37] introduced a structure called *mapping between tree graphs* (Figure 2*b*). Based on the notion of *trace* between Wagner and Fisher strings [42], a *mapping* is intuitively a description of how a sequence of edit operations transforms T_1 into T_2, ignoring the order in which the edit operations are applied. A mapping M is a set of ordered pairs (v_1,v_2) of vertices from $T_1 \times T_2$ and we say that v_1 and v_2 are images of one another. The set of vertices of T_1 (resp. T_2) which are not in a pair of M is denoted by $\overline{M_1}$ (resp. $\overline{M_2}$). Note that M is a set of pairs of vertices while $\overline{M_1}$ and $\overline{M_2}$ are sets of vertices. The set of all possible mappings from T_1 to T_2 is denoted by $\mathbf{M}$.

According to the definition of elementary costs, we can assign a cost to each mapping M:

$$\tilde{a}(M) = \sum_{(v_1,v_2) \in M} d(v_1,v_2) + \sum_{v_1 \in \overline{M_1}} d(v_1,\varepsilon) + \sum_{v_2 \in \overline{M_2}} d(\varepsilon,v_2) \quad (2)$$

The relation between a *trace* and a *sequence* of edit operations has been made explicit by Wagner and Fisher [42]. This result has been generalized for *mappings between ordered tree graphs* [37] and *unordered tree graphs* [43,45]. Mappings are related to sequences of edit operations by the following two propositions [43,45].

Property 1: Given S a sequence of edit operations from T_1 to T_2, there exists a mapping M from T_1 to T_2 such that $\tilde{a}(M) \leq \tilde{a}(S)$.

[1] A dissimilarity measure d over Σ is a function from $\Sigma x \Sigma$ to $\mathbf{R}^+$ such that for all a,b in Σ $d(a,a)=0$, $d(a,b)= d(b,a)$, $d(a,b)=0 \Rightarrow b=a$ (symmetry) and such that it does not necessarily respect the triangle inequality.

Property 2: For any mapping M from T_1 to T_2 there exists a sequence of edit operations such that $\tilde{a}(S) = \tilde{a}(M)$.

Based on these properties it can be shown that the dissimilarity between two tree graphs is measured as *the mapping with minimum cost*. Indeed, from property 1, we obtain:

$$\min_{S \in \mathbf{S}} \{\tilde{a}(S)\} \geq \min_{M \in \mathbf{M}} \{\tilde{a}(M)\}$$

Let M^* be the mapping with minimum cost. From property 2 arises a sequence S^* of edit operations such that:

$$\tilde{a}(S^*) = \tilde{a}(M^*) = \min_{M \in \mathbf{M}} \{\tilde{a}(M)\} \leq \min_{S \in \mathbf{S}} \{\tilde{a}(S)\}$$

Finally :

$$\min_{S \in \mathbf{S}} \{d(S)\} = \min_{M \in \mathbf{M}} \{d(M)\}$$

and :

$$D(T_1, T_2) = \min_{S \in \mathbf{S}} \{d(S)\} = \min_{M \in \mathbf{M}} \{d(M)\} \quad (3)$$

This equation shows that the computation of the edit distance between T_1 and T_2 leads us to solve an optimization problem, i.e. finding the mapping with minimum cost over $\mathbf{M}$.

However, when comparing plant architectures, we are not interested in all possible mappings between plants. For example, we do not want to consider mappings that match the trunk of T_1 with the leaves of T_2 and the leaves of T_1 with the trunk of T_2 (Figure 3*b*). Only those mappings that preserve certain structural properties will be considered. For example, in the case of sequence alignment, Wagner's algorithm preserves the ancestor relationship between elements of the sequence. In a tree graph, a vertex v_1 is called the *ancestor* of another vertex v_2 if a path[2] exists from v_1 to v_2. For example, one entity *a*, ancestor of an other entity *b*, can only be mapped onto an entity *a'* that is an ancestor of the image *b'* of *b*. This ancestor relationship is also denoted by $v_1 \leq v_2$. Similarly to sequences, when comparing plant architectures we wish to consider only mappings that preserve the ancestor relationship (Figure 3*a*).

One of the results from Zhang [45] and Kilpelläinen [17] is that finding the optimal matching function for an unordered tree is an NP-complete problem. This means that there is no reasonable chance of a polynomial-time algorithm solving this optimization problem. Since unordered tree graphs are important in our plant comparison applications, it is necessary to change the matching function definition in order to obtain an algorithm that computes the distance between unordered tree graphs.

An intuitive idea to solve this problem was proposed by Tanaka and Tanaka [38] who introduced a distance between ordered trees to preserve

[2] a path from v_1 to v_2 is a sequence of vertices $(w_1, w_2, ...w_n)$ such that $w_1 = v_1$, $w_n = v_2$ and for each consecutive pair of vertices (w_i, w_{i+1}) in the sequence, w_i is the father of w_{i+1}.

structural properties of the tree graphs by the matching functions. Zhang [45] extended the definition from ordered trees to unordered trees. The idea is that two separate sub-trees of one tree graph should be mapped onto two separate sub-trees.

The preservation of sub-trees can be formalized using the notion of least common ancestor. In a tree-graph, the *least common ancestor* of v_1 and v_2, denoted by $lca(v_1, v_2)$, is a common ancestor of v_1 and v_2 such that every common ancestor w of v_1 and v_2 satisfies $w \leq lca(v_1, v_2)$. For any vertex pair (v_1,v_2) of a mapping, we define a branching system with reference to their least common ancestor (Figure 3*c*). Descendants of the least common ancestor (including the least common ancestor itself) represent the branching system B_1. The images of v_1 and v_2 define another branching system B_2. The new constraint implies that: *any vertex in branching system B_1 can only be mapped onto branching system B_2*.

Mappings that preserve ancestor relationship and tree separation are called valid mappings. A *valid mapping M* is a set of ordered pairs (v_1,v_2) of vertices satisfying :

$v_1 \in T_1, v_2 \in T_2$, and for any pair (v_1,v_2), (w_1,w_2), (u_1,u_2) in M

(4) $$v_1 = w_1 \Leftrightarrow v_2 = w_2$$

(5) $$v_1 \leq w_1 \Leftrightarrow v_2 \leq w_2$$

(6) $$lca(v_1, w_1) < u_1 \Leftrightarrow lca(v_2, w_2) < u_2$$

Condition (5) expresses ancestor relationship conservation and condition (6) expresses a conservation of branching systems. The set of valid matching functions is denoted by $\mathcal{M}_v$. We can now define a dissimilarity measure between T_1 and T_2 as:

(7) $$D(T_1, T_2) = \min_{M \in \mathcal{M}_v} (\tilde{a}(M))$$

Zhang showed that the dissimilarity measure is a distance[3]. [43] According to this definition, Zhang [43,44] proposed an algorithm with bounded complexity for solving the optimization problem (7) which consists of finding a valid matching function with minimum cost. To improve the analysis of the algorithm output and consider new extensions, the computation of matching lists, i.e. the computation of mapped vertices, has been developed in [7].

The algorithm described by Zhang [43,44] uses a recursive expression for calculating distances between sub-trees of T_1 and T_2 (detailed in [7]). This algorithm solves the problem of computing $D(T_1, T_2)$ in polynomial time. Figure 4 illustrates the computation time in relation to the size of the tree graphs.

[3] This means that *D* is *a dissimilarity measure which respects the triangle inequality*.

4. The local cost function

As described in [18,28,43] and the previous section, if a distance measure is to be determined between sequences or tree graphs based upon edit operations, it is necessary to consider an elementary distance between the components of the sequences or tree graphs. In the case of plant comparison, a local distance (called the local cost function) assigns to each pair of entities (v_1,v_2) of two plants T_1 and T_2 (represented by two tree graphs), a non-negative real number (called a cost) for deleting e_1, for inserting v_2, and for changing v_1 into v_2. There are several possible methods for quantifying the difference between any two plant elements depending on the aim of the application.

A simple cost function used for comparing elementary entities is based on a binary distance called a Levenstein's distance [22]. In this case, a *null* cost is assigned to any changing operation and a cost of *one* to any insert-delete operation. A local cost defined in this way does not take into account the nature of the entities, so the distance is independent of the entities involved in the operation. A distance based on such a local cost function only involves the topological structure of plants and is called a *topological cost*.

This binary distance can be refined by using entity attributes such as length, diameter, types, etc., and defining a distance in this space. We will suppose that, for each elementary entity v of T_1 and T_2, precisely n attributes $a_1(v)$, $a_2(v)$,…, $a_n(v)$ are defined which may have symbolic or numerical values. In cases of multiple numerical attributes (n>1), it is necessary to homogenize the attribute dynamics so that they have a comparable importance in the definition of the metric. The *standardization* [15] of data consists of calculating the mean value m_i of each variable a_i and then computing for each plant T a measure of the dispersion of this variable. Traditionally, the standard deviation is used:

$$s_i = \sqrt{\frac{1}{n-1} \sum_{v_k \in T_1 \cup T_2} (a_i(v_k) - m_i)^2} \tag{8}$$

Let us assume that s_i is not zero (otherwise the variable f_i is a constant). The standardized measurements are thus defined by:

$$f_i(v_k) = \frac{a_i(v_k) - m_i}{s_i} \tag{9}$$

For numerical attributes, the elementary distance between two entities (v_1,v_2) is a metric distance in n-dimensional space, and in practice this distance is often computed as the Manhattan distance:

$$d(v_1, v_2) = \sum_{i=1}^{n} \left| f_i(v_1) - f_i(v_2) \right| \tag{10}$$

The insert-delete cost can be defined in several ways, provided that equation (1) is satisfied. For example, the insert-delete cost may be chosen to be proportional to the sum of the absolute values of the attributes:

(11) $$d_{ins}(v_1) = \mu \sum_{i=1}^{n} \left| f_i(v_1) \right| \text{ and } d_{del}(v_2) = \mu \sum_{i=1}^{n} \left| f_i(v_2) \right|$$

In order to ensure that such a local cost satisfies equation (1), μ must be a real number greater than or equal to 1.0. With such a local distance, the insert-delete cost for each entity is directly dependent upon its nature. Another way to define the insert-delete cost is to render it proportional to the absolute difference between the maximum and the minimum values of the attributes:

(12) $$d_{ins}(v_1) = d_{del}(v_2) = \mu \sum_{i=1}^{n} \left| \max_{v \in T_1 \cup T_2} (f_i(v)) - \min_{v \in T_1 \cup T_2} (f_i(v)) \right|$$

In order to ensure that such a local cost satisfies equation (1), μ must be a real number strictly greater than 0.5 [7]. Both the insert and delete costs are the most widely used in real and theoretical applications of this method [18].

Only a finite number of symbols are available for symbolic attributes. The distance between entities is defined as the distance between the different symbols. In practice the user must construct a cost matrix between these symbols. In Figure 5, T_1 and T_2 are two theoretical plants. A symbolic attribute, called a label, taken from $\{a,b,c,d\}$ is given to each entity. Table I indicates the current heuristic costs used when comparing these labels. If an entity with a given label is inserted or deleted, the assigned cost is shown in the *Null* column. The changing cost between two elementary entities relies on the comparison of their labels which is indicated in the corresponding square. In our example, the cost of comparing entity 1 and entity 2 of type *a* and *b* respectively, is 10. Thus, plants T_1 and T_2 are considered different while in a topological sense they are identical. With such local costs, the distance between the plants not only takes into account the topological structure of plants but also the information contained in the plant description.

Both types of attributes can be mixed within an appropriate local distance. Let $f_1, f_2, ..., f_k$ be k numerical attribute functions and let $f_{k+1}, f_{k+2}, ..., f_n$ be l symbolic attribute functions. According to the previous discussion, l cost matrices must be constructed that define, for each symbolic attribute, the distance between symbols. Thus, for each pair of entities (v_1, v_2) of T_1 and T_2 and for each symbolic attribute f_i, there exists a cost $c_i(v_1, v_2)$ for changing the symbol $f_i(v_1)$ into $f_i(v_2)$. In the most general form, a local distance is expressed as follows:

(13) $$d(v_1, v_2) = \sum_{i=1}^{k} \left| f_i(v_1) - f_i(v_2) \right| + \sum_{i=k+1}^{n} c_i(v_1, v_2)$$

The local cost function and the insert-delete cost are chosen depending on the application. The effect of this choice is discussed in the next section.

5. Effect of comparison parameters

The distance between plants depends on two main parameters: the topological structure of the plants and the local distance between entities. The effect of both parameters is analyzed hereafter using several sets of theoretical

plants represented in Figure 6 and Figure 7. In each set of plants (S_1), (S_2), (S_3) and (S_4), each pair of plants was compared by the algorithm using an appropriate local distance. A matrix of the distances between plants was thus obtained and these matrices were studied and analyzed depending on the application.

5.1. Effect of topological structure

Two topological structures may be different because of two major factors: their number of entities and the organization of the connections between their entities. These differences between two topological structures were evaluated separately using a *topological cost* which gives results independent of the nature of the entities.

Effect of the number of entities. The effect on the comparison of the difference in the number of plant entities was studied. A reference plant T_1 made up of ten elementary entities was constructed. One set of theoretical plants (S_1) was generated by decreasing or increasing the number of entities on each axis of the reference plant. A set of fifteen plants was thus obtained with between six and five hundred entities. A second set (S_2) of twelve plants was defined using the same method for another reference plant T_2. Figure 6 shows the distances from the plants of (S_1) and (S_2) to the reference plant T_1. When the difference in the number of entities between a given plant and T_1 is large, the distance between the plants corresponds to the difference in their number of elementary entities. Thus, the method proposed in this paper provides interesting information only for plants with a comparable number of entities.

Effect of connection between entities. If two plants have an equal number of entities, their topological structure may still differ because of the organization of the entity connections. To study this factor, we built two sets of seven theoretical plants containing ten elementary entities in their decomposition. The first set of plants (S_3) contains seven plants and is sorted according to the similarity of each plant to the reference plant T_1 (Figure 7*a*). The second set gives an example of seven theoretical plants (Figure 7*b*) with a null topological distance between each other but which are geometrically different. In (S_4) each plant is again composed of ten entities. Plants T_1 and T_2 have identical topological structures but different spatial arrangements. Plant T_2 is the mirror-image of T_1, i.e. both plants have the same branching systems but in a symmetric position with respect to a vertical axis. The algorithm gives a null distance between them:

$$D(T_1, T_2) = 0 \tag{14}$$

The spatial ordering of the children of a given entity is not taken into account by the method. Plants T_3, T_4, T_5, T_6 and T_7 have identical topological structures but differ with respect to the types of connections between their entities. The algorithm based on a topological cost does not distinguish the different types of connections ('+' or '<') between two entities. However, to make such a distinction with the algorithm, an attribute must be associated with each entity representing the connection relation between the entity and its father. A local cost depending on this attribute would thus take into account the type of connections between entities. These connections often influence the geometrical disposition of

the entities. For example, a series of entities connected by a link '<' is an axis which often can be represented as a straight segment. Such a local cost allows us to account for part of the geometric description of the entities in plants.

5.2. Effect of the local cost function

In the previous section, we compared the topological structures of plants without knowing the nature of the entities. This nature can be taken into account by defining a local distance based on the attributes of the entities. The local cost function may vary by two major parameters, the choice of the entity features (e.g. length, diameter) and the insert-delete cost. Different changing costs are defined as shown in (11) and (13) to evaluate the influence of attributes. The different values of the insert-delete cost are discussed later.

Effect of attributes. Set (S_3) was used to study the effect of different local cost functions (Figure 7*a*). Each entity of each plant was associated with several attributes such as length, diameter, number of internodes. For each attribute f_i ($i \geq 1$), a local cost function was defined as follows:

$$d_i(v_1, v_2) = \left| f_i(v_1) - f_i(v_2) \right| \tag{15}$$

$$d_{ins,i}(v_1) = f_i(v_1) \text{ and } d_{del,i}(v_2) = f_i(v_2) \tag{16}$$

The above algorithm was used to compare plants with the different local cost functions, including a Levenstein's distance denoted by d_0 [22]. For each d_i ($i \geq 0$), a distance matrix M_i, with a size 7x7, was obtained. The individual elements of M_i, $D_i(T_k, T_l)$, are computed according to the definition of the distance given by (7). In order to compare distance matrices, each individual element $D_i(T_k, T_l)$ is divided by the distance between T_1 and T_8 (this pair being arbitrarily chosen). These normalized distances are denoted by D_i^n:

$$D_i^n(T_k, T_l) = \frac{D_{f_i}(T_k, T_l)}{D_{f_i}(T_1, T_8)} \tag{17}$$

For each f_i, the mean distance to the reference plant T_1, denoted by $\overline{D_i^n}(T_1)$, is computed. Figure 8 gives the value of $\overline{D_i^n}(T_1)$ in different cases. It can be observed that for attributes showing marked variability, such as the length or diameter entities, the mean distance to T_1 is very different from the topological matching mean distance. On the other hand, for attributes such as the number of internodes per growth unit, which are roughly constant over the sets of entities, the computed mean distance is far closer to the topological computation mean distance.

Effect of insert-delete cost. In equations (11) and (12) several types of insert-delete cost were presented for each local cost function based on a given attribute, that were constant or dependent upon each entity. Set (S_3) is used to study the changes in the distance when the insert-delete cost varies. Hereafter,

only the length attribute is considered (the results are similar for other attributes) and the local cost was defined as in (10):

$$d(v_1, v_2) = \left| length(v_1) - length(v_2) \right| \tag{18}$$

$$\begin{cases} d_{ins}(v_1) = \mu_1 \times length(v_1) \\ d_{del}(v_2) = \mu_1 \times length(v_2) \end{cases} \tag{19}$$

$$\begin{cases} d_{ins}(v_1) = \mu_2 \times \left| \max_{v \in T_1 \cup T_2} (length(v)) - \min_{v \in T_1 \cup T_2} (length(v)) \right| \\ d_{del}(v_2) = \mu_2 \times \left| \max_{v \in T_1 \cup T_2} (length(v)) - \min_{v \in T_1 \cup T_2} (length(v)) \right| \end{cases} \tag{20}$$

Both coefficients μ_1 and μ_2 are real numbers which vary from 0.5 to infinity and 1.0 to infinity respectively [7]. We obtained different distance matrices for each value of μ_1 and μ_2. These matrices were normalized as explained in the previous section and the mean distance to the reference plant $\overline{D_i^n}(T_1)$ was considered. Figure 8 presents the evaluation of the mean distance to the reference plants when μ_1 is increasing (the same results can be observed with μ_2). When the cost for inserting or deleting increases, the mean distance to a reference plant decreases to a limit value equal to the mean distance to a reference tree when considering topological cost. This limit value is always obtained rapidly and depends on the attribute and the chosen insert-delete cost. If μ_1 or μ_2 are infinite, the normalized distance is equivalent to the normalized distance in the topological case. On the other hand when μ_1 or μ_2 are equal to a minimum value, the effect of the insert-delete cost on the result reaches a maximum.

6. Applications

This section briefly illustrates the use of the comparison method in different application contexts. The comparison algorithm discussed in this paper is used as a means to compare the phenotypic expression of plants. To stress the generic character of this method, three examples have been selected for comparing plants with different degrees of taxonomic genetic distances: the first example illustrates the definition of a distance between groups of plants corresponding to different growth strategies and is based on a comparison of ideal individuals representing the different groups. The second example illustrates the definition of a distance between individuals of a given genus, but with different species. Finally, the third example sketches out the application of the method in the comparison of hybrid individuals obtained by crossing two fruit tree varieties. Each application outlines different aspects of the comparison algorithm.

From a practical point of view, the user of the comparison algorithm must first define a local distance between elementary entities. This distance is defined using either real or symbolic attributes of entities. The comparison algorithm can then be used in two different contexts: either to assess the architectural variability of a set of plants or to carry out a piece-by-piece comparison between two plants.

When used for sets of plants, the algorithm produces distance matrices that can be analyzed by classical clustering methods, e.g. [15]. For pairs of plants, the algorithm outputs a list of all the matched entities. A detailed analysis of the matched subparts of the plants can then be realized.

Distance between architectural models. In the 1970's, Hallé *et al.* [12,13] proposed to identify a finite number of growing strategies characterizing the development of tropical plants. Each growing strategy is identified by a growth pattern, called an architectural model, defined by a combination of a limited set of morphological features [1]: *the growth type* (rhythmic or continuous growth), *the branching pattern* (presence or absence of vegetative branching, terminal or lateral branching, monopodial or sympodial branching, rhythmic, continuous or diffuse branching), *the morphological differentiation of axes* (orthotropy or plagiotropy) and *the position of sexuality* (terminal or lateral). For example, Corner's model corresponds to unbranched plants with lateral inflorescences. Up to 23 different models were thus identified corresponding to different combinations of morphological features. Using these concepts, Hallé and Oldeman [12] discussed the relationships between these models reflecting the architectural proximity of certain groups: for example Prévost's model and Leeuwenberg's model are claimed to be close since Prévost's model derives from Leeuwenberg's model by linear indefinite repetition [12]. Recently, Robinson [31] attempted to formalize the combination of these morphological characters by introducing an appropriate coding strategy which underlines the model similarities. For instance, Massart's model, coded by the chain of symbols $(O)r(P)$, is close to Cook's model coded by $(O)c(P)$ (where (O) symbolises an orthotropic trunk, r and c respectively represent rhythmic and continuous branching and (P) represents plagiotropic branching). According to Robinson, this formalism defines "*an appropriate symbolism that would give a framework within which relationships between the models could be explored*".

In the following application, we show how the proposed comparison algorithm can be used as a new method for comparing architectural models. We selected 12 theoretical plants representing 12 different models which can be easily modeled as tree graphs. The plants were defined with the same number of entities. Fruit position and axis orientation were described by corresponding attributes associated with each entity. Continuous branching was represented by the presence of one branch on each entity of the trunk, and rhythmic branching was symbolized by two branches on regularly spaced entities of the axes to represent branch whorls. The growth type was not represented here. A local cost was defined depending on axis orientation, fruit position and father-son relationships for each entity with the attributes having identical weights. The 12 plants were compared providing a matrix distance between "models". The distance between the plants was consistent with the used of a clustering algorithm. The taxonomy tree [2] output by this clustering technique is a tree whose terminal vertices represent the architectural models and the non–terminal vertices represent the distance between the models contained in the sub-trees (Figure 9). Three clusters can be identified: A Holtum's cluster containing Corner's and Chamberlain's model which is defined by a monopodial or sympodial trunk without branches, a

Leeuwenberg's cluster characterised by a sympodial branching sequence or a true dichotomy, and an intermediate cluster which contains models such as Massart's, and Roux's models (In another interpretation, Scarronne's model could be isolated in a fourth cluster).

Taxonomy trees between different species or genera usually reflect a genetic distance, e.g. [36]. The comparison algorithm produces another type of taxonomy tree, reflecting a phenotypic distance between groups of plants.

Clustering of pine families. The piece-by-piece plant comparison algorithm presented here was also used to compare a set of five Pinus nigra and five Pinus halepensis, 8-years old, described at growth unit scale, and obtained from simulation, (Figure 10). Three global variables were associated with each tree, namely the mean length of the tree growth units, the number of tree components and the number of branches along the trunk. These variables characterize different aspects of the tree morphology. A distance between two trees was defined for each global variables corresponding to the difference between the values for this variable in the two trees. Three matrices were computed for the set of 10 trees, corresponding to these 3 distances. Finally, a fourth matrix distance was computed using the plant comparison method presented above and Levenstein's distance.

Then, for each matrix, a classical clustering method [15] was applied to automatically separate the set of 10 pines into two clusters. We compared the obtained clusters with the original pine families. We then computed a recognition rate corresponding to the number of individuals correctly classified for the 10 individuals. Figure 11 shows that the recognition rate may vary markedly depending on the considered global variable and that the highest recognition rate (100%) was obtained for the topological comparison. This suggests that plant architectures cannot always be reduced to global variables in applications using plant architecture comparison. A piece-by-piece comparison may in fact be necessary.

Detailed comparison of hybrids. This application is intended to illustrate another aspect of the comparison algorithm output. After a piece-by-piece comparison, the algorithm provides the optimal sequence of edit operations found. The corresponding mapping between the plant entities can be observed using three-dimensional plant reconstruction [9]. Coloring tools used for 3-D representation provide a feed-back on the detailed matching between elementary tree entities. This type of analysis showed that some local similarities between two plants can appear in a global comparison. Let us consider for example the mapping resulting from the comparison of two apple tree hybrids measured at internode scale (Figure 12). The parts of the plants shown with identical colors have been mapped onto each other by the comparison algorithm and entities in black have been inserted or deleted. This mapping reveals an interesting similarity between the two hybrids: the trunk of T_1 (in red) is more similar to the (red) branching system of T_2 than to the trunk of T_2. This suggests that the differentiation sequence of the meristem which created the T_1 trunk is similar to the differentiation sequence followed by the meristem that created a T_2 branch. The biologist can use such results to orient his interpretation of the biological

phenomena. In such applications, the comparison method gives the biologist a quantitative overview of the similarity between two plants and a qualitative outline of the similar subparts of both plants.

7. Perspectives

This paper develop a technique which opens new perspectives n plant architecture modelling. Motivated by these early results, we considered extending the algorithm to multiscale structures by adding a new constraint to take into account all information contained in the plant description. In order to express both the modularity and multiscale nature of plant structures and to define a comparison method according to the topological structure of plants, we used a formalism based on multiscale tree graphs [8]. Another application of this algorithm consists of identifying a branching system in a plant and then proposing an automatic labeling method of a set of plants.

The differences between this analytical method and other methods for comparing plants need further investigation. The definition of a distance between plants highlights some general aspects concerning plant comparison are pointed out: clustering problems, automatic labeling of plant structure and, above all, the evaluation of simulated plants. These methods will be a useful and essential tool to improve plant simulation techniques.

8. Conclusion

In this paper we propose an analytical methodology for comparing plants at a macroscopic scale. Such a method gives a strong meaning to the concept of similarity between plants. The piece-by-piece tree comparison presented here was tested on various plant databases (measured apple trees, simulated and real pines).

This work is part of a project to develop quantitative evaluation tools for plant similarity. Such a tool gives a point of view different from that of global methods and uses the topological structure of plants.

9. Acknowledgements

We would like to thank Y. Caraglio for providing us with the database of simulated pines, and for fruitful discussions, E. Costes for providing us with the hybrid database and for her help in performing the detailed comparison analysis, and S. Henton for reviewing the English in preliminary versions of the manuscript.

10. References

[1] D. Barthélémy, C. Edelin and F. Hallé, Architectural concepts for tropical trees, 1989, 89-100, *Chapter in a book :* Tropical forests : Botanical dynamics, speciation and diversity, Academic Press London

[2] J.-P. Barthélemy and A. Guénoche, Les Arbres et Les Représentations Des Proximités, 1988, Collection Méthode + Programmes, Masson, Paris.

[3] E. Costes, H. Sinoquet, C. Godin and J. J. Kelner, 3D digitizing based on tree topology : application to study the variability of apple quality within the canopy, Acta Horticulturae [1998]

[4] P. deReffye, E. Elguero and E. Costes, Growth units construction in trees: a stochastic approach, Acta Biotheoretica 39 [1989] 325-342

[5] P. deReffye, T. Fourcaud, F. Blaise, D. Barthélémy and F. Houllier, A Functional Model of Tree Growth and Tree Architecture, Silva Fennica 31 [1997] 297-311

[6] J. Edmonds and R. M. Karp, Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems, Journal of the Association for Computing Machinery 19 [1972] 248-264

[7] P. Ferraro, Une méthode de comparaison structurelle d'arborescences non ordonnées, *technical report,*Cirad [1998], Montpellier

[8] C. Godin and Y. Caraglio, A multiscale model of plant topological structures, Journal of Theoretical Biology 191 [1998] 1-46

[9] C. Godin, E. Costes and Y. Caraglio, Exploring plant topological structure with the AMAPmod Software: an outline, Silva Fennica 31 [1997] 355-366

[10] C. Godin, Y. Guédon, E. Costes and Y. Caraglio, Measuring and analyzing plants with the AMAPmod software, 1997, 63-94, *Chapter in a book:* Advances in computational life sciences, Vol I: Plants to ecosystems, CSIRO, Australia

[11] Y. Guédon and E. Costes, A statistical appraoch for analyzing sequences in fruit tree architecture, Fifth international symposium on computer modelling in fruit research and orchard management [1998], *Proceeding*, Acta Horticulturae, ISHS

[12] F. Hallé and R. A. A. Oldeman, Essai sur l'architecture et la dynamique de croissance des arbres tropicaux, 1970, Paris,

[13] F. Hallé, R. A. A. Oldeman and P. B. Tomlinson, Tropical trees and forests. An architectural analysis, 1978, Springer Verlag New York Heidelberg Berlin

[14] J. S. Hanan and P. Room, Practical aspects of plant research., 1997, Australia,28-43,*Chapter in a book*

[15] L. Kaufman and P. J. Rousseuw, Finding groups in data, 1990, Wiley Series in Probability and Mathematical Statistics

[16] J. D. Kececioglu and E. W. Myers, Combinatorial algorithms for DNA assembly, Algorithmica 13 [1995] 7-51

[17] P. Kilpelläinen and H. Mannila, The Tree Inclusion Problem, 1 [1991] 202-214, *Proc. Internat. Joint Conf. on the Theory and Practice of Software Developmen*t (*CAAP'91*)

[18] J. B. Kruskal, An Overview of Sequence Comparison, 1983, 1-44 ,*Chapter in Book:* Addison-Wesley Publishing Company Inc.

[19] M. Küppers, Carbon relations and competition between woody species in a Central European hedgerow, Oecologia 66 [1985] 343-352

[20] W. Kurth, Growth grammar interpreter GROGRA 2.4: A software for the 3-dimensional interpretation of stochastic, sensitive growth grammars in the context of plant modelling, *technical report,* Forschungszentrum Waldoekosysteme der Universitaet Goettingen [1994]

[21] S. LeDizès, P. Cruiziat, A. Lacointe, H. Sinoquet, X. LeRoux, P. Balandier and P. Jacquet, A Model for Simulating Structure-Function Relationships in Walnut Tree Growth Processes, Silva Fennica 31 [1997] 313-328

[22] A. Levenstein, Binary codes capable of correcting deletions, insertions and reversals, Sov. Phy. Dokl. 10 [1966] 707-710

[23] S. E. Levinson, A. E. Rosenberg and S. E. Levinson, Evaluation of a word recognition system using syntax anlysis, Bell Syst. Tech. Journal 57 [1978] 1619-1626

[24] S.-Y. Lu, A tree-to-tree distance and its application to cluster analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence 1 [1979] 219-224

[25] R. Mech and P. Prusinkiewicz, Visual models of plants interacting with their environment., SIGGRAPH 96 Conference Proceedings [1996] 397-410

[26] L. Miclet, Méthodes Structurelles Pour la Reconnaissance Des Formes, 1984, 61, bd Saint-Germain - 75005 Paris, 184, *Book*

[27] A. S. Noetzel and S. M. Selkow, An Analysis of the General Tree-Editing Problem, 1983, 237-252*,* Addison-Wesley Publishing Company Inc

[28] A. Ohmori and E. Tanaka, A unified view on tree metrics, Syntactic and Structural Pattern Recognition 45 [1988] 85-100

[29] F. P. Preparata and R. T. Yeh, Introduction to Discrete Structures for Computer Science and Engineering, 1973,

[30] P. Prusinkiewicz and A. Lindenmayer, The algorithmic beauty of plants, 1990, New York,

[31] D. F. Robinson, A symbolic framework for the description of tree architecture models, Botanical Journal of the Linnean Society 121 [1996] 243-261

[32] S. M. Selkow, The tree-to-tree editing problem, Information processing letters [1977] 184-186

[33] H. Sinoquet, P. Rivet and C. Godin, Assessment of the three-dimensional architecture of walnut trees using digitizing, Silva Fennica 31 [1997] 265-273

[34] E. Sobel and H. Martinez, A multiple sequence alignment program, Nucleic Acid Res. 12 [1986] 75-88

[35] R. P. Stanley, Enumerative combinatorics, 1986, Monterey vol. 1

[36] D. L. Swofford and G. J. Olsen, Phylogeny reconstruction, 1990, Sunderland, Massachussets, USA, 411-501, *Chapter in a book:* Molecular Systematics, Sinauer Associates, Sunderland, Massachussets, USA

[37] K.-C. Tai, The tree-to-tree correction problem, Journal of the Association for Computing Machinery [1979] 422-433

[38] E. Tanaka and K. Tanaka, The tree-to-tree editing problem, Internat. Jour. Pattern Recognition And Artificial Intelligency 2 [1988] 221-240

[39] E. Tanaka, T. Toyama and S. Kawai, High speed error correction of phoneme sequences, Pattern Recognition 19 [1986] 407-412

[40] R. E. Tarjan, Data Structures and Network Algorithms, 1983, 97-111, *Chapter in a book,* Regional Conference Series in Applied Mathematics, Murray Hill, New Jersey USA

[41] R. E. Tarjan, Data Structures and Network Algorithms, 1983, 85-96, *Chapter in a book:* Regional Conference Series in Applied Mathematics, Murray Hill, New Jersey USA

[42] R. A. Wagner and M. J. Fisher, The string-to-string correction problem, Journal of the Association for Computing Machinery 21 [1974] 168-173

[43] K. Zhang, A new editing-based distance between unordered trees, 4th CPM'93 Padala (Italy) [1993] 254-265

[44] K. Zhang, A constrained edit distance between unordered labeled trees, Algorithmica 15 [1996] 205-222

[45] K. Zhang and T. Jiang, Some MAX SNP-hard results concerning unordered labeled trees, Information Processing Letters 49 [1994] 249-254

11. Table of captions

Figure 1: Plant topology described at different scales, i.e. in terms of

(*a*) branching systems,
(*b*) growth units,
(*c*) internodes,
and represented as rooted tree graphs (on the right hand side)

Figure 2: Mapping from one tree graph T_1 onto another tree graph T_2.

(*a*) The five edit operations used to transform T_1 into T_2.
(*b*) Resulting matching function from T_1 onto T_2 where black vertices represent the inserted or deleted vertices.

Figure 3: Allowed and forbidden matching functions in tree graph comparisons:

(*a*) preservation of ancestor relationship,
(*b*) non-preservation of ancestor relationship,
(*c*) preservation of branching system,
(*d*) non-preservation of branching system.

Figure 4: Computation time according to the size of the tree graphs (run on a SGI5000 Silicon graphics station)

Figure 5: Comparison by label.

(*a*) Theoretical tree graphs with labeled entities. For each entity, *a* represents a large length and a large diameter, *b* represents a small length and a large diameter of the entity, *c* represents a small length and a small diameter, and *d* represents a large length and a small diameter. These values are graphically represented on the biological representation (*b)*.
(*c*) Distance from T_1 to T_2 and T_3 as computed by the algorithm.

Figure 6: Topological comparison

(*a*) Sets S_1 and S_2 of theoretical plants built from T_1 and T_2**.**

(*b*) Distance from plants of S_1 and S_2 to reference plant T_1 on a logarithmic scale.

Figure 7: Two sets of theoretical plants:

(*a*) Plants of S_3 have different topologies. In the figure, plants are sorted according to their topological similarity to the reference plant T_1.
(*b*) Plants of S_4 have a similar topology and different geometry.

Figure 8: Mean distance to the reference plant T_1

(*a*) Value of the mean distance using several local costs,

(*b*) Changes in the mean distance according to the changes in the insert-delete cost value, the gray line represents the mean distance with a Levenstein's distance and its asymptote.

Figure 9: Taxonomy tree whose terminal vertices correspond to architectural models and the non–terminal vertices represent the distances between the models appearing on the leaves

Figure 10: Three individuals from each pine set: Pinus halepensis on left-hand side and Pinus nigra on right-hand side.

Figure 11: Recognition rate of the clustering algorithm for different definitions of the distance between individual pine trees:

(*a*) distance defined as the difference of growth units,
(*b*) ditto but using the total number of branches on the trunk,
(*c*) ditto but using the number total number of growth unit,
(*d*) distance defined by the piece-by-piece comparison algorithm using a Levenstein's distance.

Figure 12: Detailed analysis:

Each internode of the apple tree on the left-hand side is colored according to its order: red for order 1, blue for order 2 and yellow for order 3. Black denotes deleted internodes. The matched entities of the second apple tree are colored with the same color as their image and the inserted entities are colored in black. A similarity between the trunk of the first apple tree and a branching of the second plant (in red color) is outlined by the matching.

Table I: Heuristic local distance between label *a*, *b*, *c* and *d*.