Back to the trees: Identifying plants with Human Intelligence

Simon Castellan, Jos Käfer, Eric Tannier

Back to the trees: Identifying plants with Human Intelligence

Simon Castellan
Inria, Centre de l'université de Rennes; Univ Rennes
France
simon.castellan@inria.fr

Jos Käfer
Université de Lyon, Université Lyon 1, CNRS, Laboratoire de Biométrie et Biologie Evolutive UMR 5558, Villeurbanne, France; DIADE, Université de Montpellier, IRD, CIRAD, Montpellier, France; ISEM, Univ Montpellier, CNRS, IRD, Montpellier
France
jos.kafer@cnrs.fr

Eric Tannier
Inria, Centre de Lyon ; Université de Lyon, Université Lyon 1, CNRS, Laboratoire de Biométrie et Biologie Evolutive UMR 5558, Villeurbanne
France
eric.tannier@inria.fr

ABSTRACT

We investigate a way to build a *convivial* plant identification tool halfway between the complex determination keys of botanists and the more recent but poorly explainable approaches based on AI image recognition. Our approach consists of a formal language to organize morphological traits and a Bayesian technique to describe plants with possible polymorphisms at all taxonomic levels, and to handle errors and uncertainties. From these structured data, automatic approaches can be designed to generate versatile *determination keys*, i.e. decision trees, which are otherwise tedious to design by hand.

ACM Reference Format:
Simon Castellan, Jos Käfer, and Eric Tannier. 2023. Back to the trees: Identifying plants with Human Intelligence. In *LIMITS '23: Workshop on Computing within Limits, June 14–15, 2023.* ACM, New York, NY, USA, 12 pages.

1 INTRODUCTION

Knowledge about our vegetal neighbors has always been a key skill for human survival, as plants provide food and medicine, among others. The activity of describing and classifying plants is thus ancient, and a science has slowly developed from it over the centuries. Taxonomic classifications were formalized by Linnaeus, with the most widely used levels being *species*, *genera*, and *families*. These classifications have been constantly modernized, recently following phylogenetic criteria and using genetic markers instead of morphological ressemblance. Yet the ways to identify plants have remained relatively constant, and continue to be used by scientists, farmers, gardeners, pharmacists and amateurs alike.

Identification can be achieved by *determination keys* [de Lamarck 1779], which are decision trees, not necessarily following the taxonomy, whose leaves are species (or varieties, or families, or genera), and nodes correspond to a morphological observation. Determination keys remain very popular today among botanists (*Flora Gallica* [Société botanique de France 2014], *Flora Europaea* [Tutin et al. 2001] for instance) as precise tools to identify an unknown plant, despite some limits: (1) they require an expertise in plant morphology; (2) they suppose the ability to answer questions concerning all the organs (and in particular the flower) even if these organs are not observable (because of seasonality for example) on the considered plant; and (3) they leave little place for errors or uncertainties, both from the descriptions and from the observations.

The popularization of computers has yielded attempts to digitize determination keys[1][2], design new ones[3], invent other ways to identify a plant than decision trees, as a flat "robot portrait" instead of the decision tree[4][5][6], or design algorithms to automatically construct a key from data [Kerner and Lebbe 2019].

Surprisingly, these attempts have yielded little formalisation of both the existing descriptions for use in computers, and the properties of the determination keys. Descriptions were mainly achieved by matrices with species as lines and attributes as columns, in which an entry is a value of one attribute for one species. Uncertainties, dependencies between attributes, polymorphism, were hardly addressed by this solution.

Research in this direction has fallen out of fashion because of several concomitant scientific and cultural revolutions. On the one hand the rise of genomics provided a more immediately formalizable data source for scientific studies than morphological descriptions. On the other hand the activity of identifying plants has been extended to a wide public thanks to the combination of progress on image recognition through deep learning and the cultural shift that made billions of people own an internet-connected smartphone. Applications identifying plants based on a single picture (iNaturalist [Horn et al. 2018], Pl@ntNet [Affouard et al. 2019], Google Lens) became very popular among amateurs, as they give an almost instant answer.

We would nonetheless like to argue that such systems, although we recognize their efficiency and usefulness, do not close definitively the identification question. Firstly, precise identification often requires a more thorough observation than what can be seen on an

LIMITS '23, June 14–15, 2023,

[1]Tela-Botanica has digitised the Bonnier key https://www.tela-botanica.org/eflore/bonnierpda/Bonnier.html
[2]https://efloras.org
[3]https://www.florenum.fr
[4]NatureGate https://luontoportti.com/fr
[5]FloraGator https://hort.ifas.ufl.edu/floragator/key_without_thome.html
[6]WeedId https://weedid.missouri.edu/weedKey.cfm

amateur picture. Secondly, the answer of such applications lacks *explainability*: it does not fully replace determination keys that connect observable features with the identification. AI systems are oracles that have to be trusted almost blindly, and they cannot provide human-understandable feedback on the identification. Furthermore, they require important infrastructure: servers to train models on, a smartphone with a camera and a dense mobile coverage of the earth's surface allowing internet connections.

Identification tools can be investigated and compared according to their *conviviality*, with a reference to Illich [Illich 1973], as well as to the connected notion of *self-obviating* systems [Tomlinson et al. 2015]. Conviviality means, in our sense, inspired by Illich, a tendency for a tool to increase the autonomy of users, and to avoid creating either dependencies to the tool, or unwished dependencies to other social groups or structures. Self-obviating tools are good examples of convivial tools: they are designed so that their usage makes themselves more and more useless. As remarked by Tomlinson et al [Tomlinson et al. 2015] an ideal plant identification tool would be self-obviating because each usage teaches to the user an association of an observation with a name. Identification keys reinforce this quality: the user associates the distinctive plant morphological traits to the name, and even if very few persons would be able to identify all species in a country without a key, its use becomes easier through time as one knows more species, and the distinctive traits at higher taxonomic levels. The explainability of a tool thus plays an important role in this property: an association is better learned if it is given a *meaning* that users can understand and remember.

We here aim to use computers in plant identification in a way that keeps this self-obviating aspect of identification keys, and at the same time render them more convivial. In this paper we revisit botanical determination keys in the light of both modern computer science and conviviality. We take advantage of formal language descriptions, bayesian statistics, and the generation of decision trees [Quinlan 1986] to show that computer science has the potential to improve determination keys without losing their interesting characteristics.

Our proposition is based on (1) a formalisation of the morphological description of species, including variabilities and uncertainties (Section 3) (2) an algorithm generating determination keys (Section 4) for formalized data. In Section 5, we present our ongoing attempts at gathering data.

2 FORMALISING PLANT MORPHOLOGY

Descriptions of families, genera and species can be found in *floras*, books compiling a list of existing species in a particular region (e.g. province, country, mountain range, continent). These books are written in natural language which is more or less codified and thus the information they contain can be difficult to use in an automatic way. We first summarise some difficulties in describing plants in natural language, and then explore some difficulties in translating them into a formal language.

2.1 Describing plants

A description is a list of properties that should be *observable* without too much technological involvement (e.g. *the leaf is directly attached to the stem*). Properties have to be **inclusive** enough to capture the variation within one species: they are based on observations of several individuals of the species, in different locations, at different times of the year, and at different stages of growth. They also have to be **exclusive** enough to differentiate it from other species. They contain discriminating aspects of the plant. For instance, few floras mention that leaves are green as this is implied by default, and is not a very helpful property to identify a plant.

Figure 1: *Potamogeton gramineus*, photo from Wikipedia

Within a flora, descriptions tend to show some uniformity: reusing the same terminology and the same properties across different species exhibiting similar morphology, whenever possible.

For example, the description of *Potamogeton gramineus*, of which a photo is shown on Figure 1, in the Flora Europaea [Tutin et al. 2001] is:

- "Floating leaves (if present) up to 7×3 cm, elliptical or ovate-elliptical, cuneate or rounded at the base, opaque; petiole often longer than the lamina. Submerged leaves (at least the lower) up to 8×3 cm, sessile, narrowly elliptic-oblong to narrowly elliptical or oblanceolate-oblong, cuneate at the base, acute or acuminate, minutely denticulate at least when young, with regularly ascending secondary veins (occasional leaves more or less reduced to phyllodes). Stipules conspicuous, herbaceous. Peduncles thickened upwards."

2.2 Challenges in formalising descriptions

An immediate idea to transform floras into a formal language is to transform a natural language property like the one extracted from the description above "Submerged leaves (at least the lower) up to 8×3 cm, sessile" into an attribution of values to variables (the *attributes*): Attribute *Attachment of the leaf* has value *sessile* (attached to the stem), attribute *length* has value up to 8, width has value up to 3, *position* has value "submerged*.

A first challenge is the familiarity with the botanical vocabulary (sessile, stipule, are not necessary common). For this we use a set of drawings showing, at each occurrence, the associated organ or

Figure 2: A botanical panel describing *Fragaria vesca*, the wood strawberry, pictured from the book by Otto Wilhelm Thomé *Flora von Deutschland, Österreich und der Schweiz* 1885, Gera, Germany, and used under Creative Commons by wikimedia commons. The species has compound leaves (each leaf consisting of three leaflets). 1,2. Flower bud at two developmental stage. 3. Flower, where petals (white), sepals (green), stamens (yellow) and pistils are visible. 4. Stamen. 5. Pistil. 6. The strawberry "fruit" consists of the expanded flower receptacle, the actual fruits (in a botanical sense) being the seeds attached to the surface of this receptacle.

Figure 3: Botanical panel of *Euphorbia helioscopia* (A) and *Euphorbia esula* (B), from the same source as Figure 2, under Creative Common licence. 1. Compound inflorescence, showing three cyathia (see 3). 2. Outer inflorescence bract. 3. What appears to be a flower is actually a compact inflorescence, called "cyathium", consisting of flowers that are highly reduced to one pistil or one stamen (5 is a cross-section). 4. Dissected involucre, consisting of fused bracts surrounding the flowers in the cyathium. 6. Male flowers, each consisting of one stamen. 7,8. Female flower consisting of one pistil (7. a horizontal cross-section of the ovary.) 9. Seed.

character. In this article we illustrate this way of taking care of neophytes by showing two botanical panels and refering to them when using a specialised vocabulary. The first one, strawberry, illustrate standard terms, with flowers, petals, sepals, stamen, pistil, fruits. The second one illustrates a specialised inflorescence composed of flowers that lack perianth (petals or sepals).

Another challenge is that from the example description above it is obvious that this transformation is not immediate. For example, there are several kinds of leaves, some submerged, some floating. This *polymorphism* (several forms for a single subject) is frequent also for non aquatic herbs, with different basal (attached to the ground part of the plant) and caulinar (attached to the aerian part of the stem) leaves. The range of quantitative traits is unprecise, "more or less reduced to phyllodes" is hardly formalized.

Polymorphism of leaves on a single plant like in the description of *Potamogeton gramineus* can be generalized to polymorphism at any level: from within a single organ (a petal can have several colors, several petals of a single flower can have different colors, petals from different flowers from the same species can have different colors). Attributing several values is thus ambiguous.

Indeed, variation is present at every level, both within a plant, within populations or species, and among species. Although we often present it in a discrete way (classification), it is mostly the manifestation of some quantitative variation. There are for instance many ways in which flowers can be organized into inflorescences, and while some are readily identified (e.g. the "flower head" of Asteraceae, the umbel of Apiaceae), there are many intermediate cases. It's like classifying colors, we have a few words (red, orange,

yellow, etc.) to describe something that appears in nature as a continuum.

This variation also makes it sometimes hard to correctly identify the organ. That the white structures that surround the daisy "flower" aren't petals but flowers themselves is relatively widely known, even outside botanical circles. Most Euphorbia species (a very large and widespread genus, see an exemplar in Figure 3) have highly reduced flowers, each consisting of one pistil or one stamen, organized in such a way that the inflorescence (called "cyathium") mimics a flower quite perfectly: it usually takes a trained botanist to know this. In some cases, determining the exact status of a structure or organ requires detailed comparative anatomical and genetic studies that have not yet been performed. E.g. petals, sepals, stamens and pistils are floral structures all thought to be more of less modified leaves, yet some species have petaloid sterile anthers and lack "true" petals, so what one calls a petal is a matter of scientific consensus that can change when the understanding of flower development changes. Indeed, botany is a science in itself, with its own vocabulary, theories and open questions.

Last but not least, the botanical knowledge is spread out in different works. They have been published at different times, in different countries and different languages, and thus use different sets of attributes and values. There have been some efforts towards standardization, but even a flora of a medium-sized country like France contains about 6000 species, and it isn't easily updated to reflect the latest accepted terminology. Besides, for older works, the terminology is not always explicitly defined, making it hard to precisely compare the information contained in different floras.

In summary, floras are a way to shape botanical knowledge, which is a first formalization step. Transforming floras into chosen categories and values can only express knowledge at the cost of sometimes arbitrary and unpredictable choices at the limits of the categories, and at the risk of a loss of part of the knowledge, which is not expressed with those categories. The process of transforming knowledge into information is always at this cost. We will try to partly tame this arbitraryness by using probabilities above the categories.

2.3 Existing formalised databases

2.3.1 Existing botanical databases. Attempts to formalise plant morphology are too numerous to be exhaustively cited, we will only consider a few recent attempts that produced publicly available data. They are often focused on particular aspects of plant morphology to allow comparisons between species, populations or individuals, and thus include traits that can be defined for (almost) all species [e.g. Sauquet et al. 2017]. More specialized databases in ecology focus on functional traits that are designed to be applicable to a large number of species, such as the "Specific Leaf Area' ' which measures the ratio of leaf surface and dry mass. While plants species differ in this trait, it is neither distinctive enough to identify species, nor easily observable without specific tools. The popular database TRY contains many of such functional traits.

2.3.2 Existing morphology descriptions. To go beyond relational data (attribute/value for each species), knowledge bases tend to be described *ontologies*. Ontologies are graphs where nodes are *concepts* (in our case: `plant`, `leaf`, `flower`, …); and edges *relations* between concepts (in our case: plants may have leaves, meaning a relation `plant` → `leaf`). We can then describe species as individuals of the concept `plant` using a logical language such as OWL [OWL 2012]. This uniform framework allows the design of tools that work on all knowledge bases using this formalism (query, statistics, …). Ontologies for plants [Jaiswal et al. 2005] focus on the anatomy of the plants and growth stages, rather than identification criteria. We also found that existing ontology frameworks were not exactly what we wanted: in our opinion, concepts and relations tend to make describing the hierarchy of attributes and values a bit heavy. Moreover, there is no standard support for probabilistic features we need, and thus we would have to probably modify an existing system anyway.

For these reasons, we have instead drawn inspiration from ontology systems but have designed something we believe to be simpler to tailor to our specific needs. Our system is designed to be compatible with OWL so that the data can be exported into a format most OWL-related software can work with.

3 A HIERARCHICAL, EVOLVABLE, PROBABILISTIC DESCRIPTION OF PLANTS

We present our proposition of structure of a morphological database of plants, suitable to construct a determination key, and by the way usable in a large spectrum of scientific activities around plants, their morphologies, functions, ecology, interactions, evolution.

3.1 Desired properties

1. **Explicitation of the schema.** The list of attributes and their values is sometimes left implicit in existing databases. We would like our *schema* to have an explicit description in a formal language easy to understand, discuss and revise.
2. **Hierarchy of attributes.** Not all species have a value for attributes. Indeed, there is a natural dependency relation between attributes that informs when an attribute is relevant for a plant. For instance, if the species is not flowering, all the attributes about flowers are irrelevant. We want a schema that makes these dependencies explicit. This helps with entering data (as we see directly which fields are relevant or not) but also with understanding the structure of the schema. Hierarchy can also encourage modularity of the schema by splitting it into subschema.
3. **Structured representation**. Most databases are a flat representation which must lose out on the structure of the flora description. We believe a more structured approach is necessary, in particular to understand *correlations between traits*. In databases, each trait is supposed to be independent, which is not always the case. Indeed, to represent species with different kinds of leaves (basal or on the stem for instance) or flowers, matrices cannot provide a detailed representation.

Guided by intuitions from programming language theory, we view schema as types of a programming language. Our method is thus:

- To define a **type theory**, i.e. a language to define and manipulate schema. We want this language to satisfy the criteria

above, but also allow us to describe the schema (1) *modularly* (easily split in subschemas) and (2) mimic the way morphology is presented in textbooks.
- To define a **denotational semantics** for this theory, that is for every type (representing a schema) T, we describe a set $\mathcal{O}(T)$ of probabilistic observations valid according to schema T. This resumes to attributing a value to an attribute with a certain probability.

3.2 Schemas as types

Our first work was towards thinking about how to formalise a list of attributes and values that are relevant for identification and used in most floras. This list is meant to be *evolvable*, as the work of actually compiling an exhaustive list adapted to all families, if at all possible, is a an endeavor spanning years.

In terms of computer science, we want to describe our schema as a type $\mathbb{P}$ in a suitable type theory, representing the structure of observable traits on plants. Traditional databases have a simple schema. Each attribute can be seen as a sum type (also known as enum types), and the overall scheme is a product (or record, or structs) of each attributes. Thus, such schema are products of sums.

To improve on this flat structure, we follow a more sophisticated approach based on algebraic types found in ML languages [Milner et al. 1997]. Their tree structure have two main advantages:

- *Hierarchy.* They can represent nicely the hierarchical structure of observable traits: flowers, then perianth, then petals, … They also make explicit the dependency between traits: petals only make sense for flowering plants.
- *Incrementality.* It makes it easy to leave some parts undescribed to later detail and replace it with a more precise description. For instance, we can start by describing the hair on the stem as a boolean before moving on to a more complex description of the hair. This is essential as we cannot wait to have a complete type before starting to describe species, and allows to easily insert new branches for describing specific families (e.g. the conifers with the needles, the grasses, or the composite flowers).

3.2.1 Algebraic types for describing plant structure. Algebraic types are used in functional programming languages. They are built from *product types* (often known as *records* in non-functional languages) and *sum types* (related to *union* or *enum* in non-functional languages). They can be seen as describing the shape of hierarchies, thus well-suited to represent morphological descriptions (plant, then flowers, then perianth, then corolla, ….).

Standard traits can be represented as simple cases of sum types:

```
color := [ red | blue | yellow | ... ].
```

To describe an organ or a component composed of several subcomponents, records can be used:

```
corolla := { color: color; number: [ 3 | 5 | 6 | ... ] }.
```

This defines a corolla has having two sub-components: its color, described by the previous `color` type, and the number of petals.

Sum types are convenient for representing parts of the description that are specific to certain species. For example, the case of flower heads can be represented as follows:

```
inflorescence-type := [ capitulum (cap-descr) | umbel | ... ].
cap-descr := {
  involucre: involucre;
  ray-flowers: [ yes | no ];
  center-flowers: [ yes | no ];
}.
```

The first line defines the type of inflorescence. This definition can be read as defining an attribute `inflorescence-type` with at least two values `capitulum` and `umbel`. The `(cap-descr)` indicates that when the value `capitulum` is selected, new attributes are unlocked, described by `cap-descr`.

Finally, as leaves of the algebraic types, we also allow quantitative traits (for representing lengths for example).

3.2.2 The formal grammar of types. To study mathematically types obtained by such constructions, we define the following grammar:

$$\begin{array}{rcll} S & ::= & S_1 \oplus \ldots \oplus S_n & \textit{Sum types} \\ & | & S_1 \times \ldots \times S_n & \textit{Record types} \\ & | & \mathscr{M}(S) & \textit{Multiple type} \\ & | & [a-b]u & \textit{Quantitative trait} \end{array}$$

In the first two lines, n can be zero leading to the empty type $\mathbf{0}$ and the unit type $\mathbf{1}$. This abstract syntax does not have names for constructors and fields, but our concrete syntax does. When we write `[yes | no]`, we mean the sub-type $\mathbf{1} \oplus \mathbf{1}$. And a record `{flower: [yes|no]; leaf:[yes|no]}` becomes $(\mathbf{1} \oplus \mathbf{1}) \times (\mathbf{1} \oplus \mathbf{1})$. For instance, written formally the type `inflorescence` above is:

$$(\text{involucre} \times \mathbf{2} \times \mathbf{2}) \oplus \mathbf{1} \oplus \ldots$$

where $\mathbf{2} := 1 \oplus 1$. Each operand to the $\oplus$ correspond to a case of `inflorescence`. Empty cases (e.g. `umbel`) become $\mathbf{1}$.

The last line is a quantitative trait: u is the unit, and $[a, b]$ is an interval describing the default probability distribution. For instance, leaf length could be represented as $[1-100]$cm.

The most surprising construct is the third lines, that we call *Multiple types*. This can be used when the sub-type S can have several forms that need independent description. It can be thought of as a type of multiset and is inspired by the exponentials in Linear Logic [Girard 1987] and cardinality constraints in OWL.

3.2.3 Defining the type: methodology. We have started writing a `plant.scheme` following this methodology, gathering information both from floras and determination keys, as well as morphological books, guided by the botanical expertise of members of the group.

As mentioned in Section 2, devising this formal scheme proves to be a arduous endeavour. First, different authors have a different way of structuring certain aspects and finding out which is the best way to formalise can be tricky. Moreover, identification traits are about delimiting certain behaviours (certain leaves are simple, other are compound), but for each such delimitation, there are always corner cases for which there is a bit of uncertainty involved in choosing the value. As a guideline in formalising, we try to limit the number of values for a trait, which means trying to split traits with a large number of values into sub-traits. This may help pushing the uncertainty to sub-traits which are less critical to the identification.

Figure 4 represents part of our current `plant.scheme`. Nodes in orange are attributes; nodes in purple values. The unlocking of attributes corresponds to arrows from values to attributes.

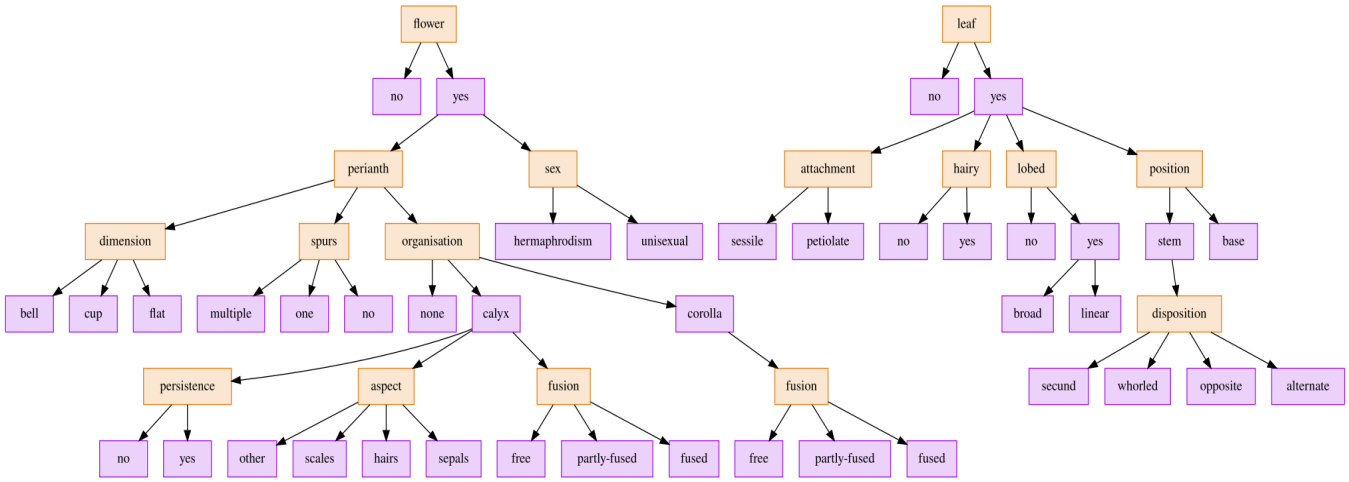

Figure 4: A part of our `plant.scheme` displayed as a tree. Note that for illustration purposes, only a subset of the attributes (orange) and their values (purple) are shown.

3.3 Probabilistic and structured description of plants

To model uncertainty in botanical knowledge and possible polymorphisms at all levels (within an organ, a plant or a population) we follow a bayesian approach. For this, we need a probabilistic description of species. In its simplest form, this means that a species description does not assign a value to every attribute, but rather a distribution of probability representing uncertainties. Obviously, translating the often qualitative uncertainty present in floras into quantitative probabilities involves a certain amount of guesstimating, but we believe that, even though the probabilities are not extremely reliable, they can be used to protect ourselves from errors in the data, and difference in appreciation, as well as representing species polymorphism evoked in Section 2.2. Moreover, as we will see, we also want to adopt a bayesian approach for identifying a species (see Section 4) and so it will fit right in.

As we will see in the next section, it is not quite enough to just have probabilistic distribution of values for each attribute, as this misses correlations between traits. Those correlations tend to be quite rare but important for a large class of species. Typical correlation include for instance structure of the leaf and position in the plant (stem or base).

3.3.1 The space of observations. From the type $\mathbb{P}$ mentioned in the previous section, we derive a structure for descriptions. Descriptions live in a space $\mathcal{O}(\mathbb{P})$. Since we want descriptions to be probabilistic, we could have taken $\mathcal{O}(\mathbb{P}) := \mathcal{D}(\mathbb{P})$ the space of probabilistic distributions over $\mathbb{P}$.

Since $\mathbb{P}$ is complex, so is describing directly a probability distribution on it. It is much easier to define a probability distribution per attribute in most cases, while using multiple types to represent correlations.

We take advantage of the inductive definition of $\mathbb{P}$ to describe $\mathcal{O}(\mathbb{P})$ by induction using simplification formulas.

- **Records**. It is well-known that there is a correspondence between $\mathcal{D}(S\times T)$ – distributions over $S\times T$ and $\mathcal{D}(S)\times\mathcal{D}(T)$ – pairs of distributions. This correspondence is not one-to-one as from left-to-right it forgets correlations between S and T. We still choose to use this correspondence here to simplify $\mathcal{O}(S\times T)$, leaving correlations to be dealt with multiple types. Thus we let:

  $$\mathcal{O}(S_1 \times \cdots \times S_n) := \mathcal{O}(S_1) \times \cdots \times \mathcal{O}(S_n)$$

  We write tuples $\mathcal{O}(S_1 \times \cdots \times S_n)$ as $\langle o_1, ..., o_n\rangle$.
- **Sum**. A distribution over a disjoint union $S \oplus T$ can be described by a distribution over S and one over T, plus a weight in $[0, 1]$ that says which side we are closer to. Thus, we let:

  $$\mathcal{O}(S_1 \oplus ... \oplus S_n) := \mathcal{D}([1, n]) \times \mathcal{O}(S_1) \times \cdots \times \mathcal{O}(S_n)$$

  Instead of writing $(d, o_1, ..., o_n)$ we write $d(1) \cdot o_1 \oplus \ldots \oplus d(n) \cdot o_n$. If $S_i = \mathbf{1}$ for all i, we use $\overline{i}$ to denote the certain i-th value of that attribute: $(\delta(i), (), ..., ())$ where $\delta(i)$ is the Dirac distribution, i.e. maps i to 1 and everything else to zero.
- **Multiple type**. The values of $\mathcal{M}(S)$ are to be thought of "multisets" of values of S, to be able to represent several forms of the same component. To be able to represent correlations, we go beyond and choose $\mathcal{M}(S)$ to be represented as probability distributions of S:

  $$\mathcal{O}(\mathcal{M}(S)) = \mathcal{D}(\mathcal{O}(S)).$$

  The distributions are written $[p_1 : o_1, ..., p_n : o_n]$.
- **Quantitative data**. For quantitative data, we just interpret them as continuous distributions over $\mathbb{R}_+$. In the implementation for now, we only support normal distributions for simplicity.

3.3.2 Representing correlations. Using multiple types, we can represent correlations. For instance, consider the following schema of leaves:

```
leaf := { position: [basal|stem]; size:[small|large]}.
```

Formally, `leaf` $:= (\mathbf{1}\oplus\mathbf{1})\times(\mathbf{1}\oplus\mathbf{1})$. Now assume we let `plant` $:= \mathcal{M}($`leaf`$)$.

We can represent a plant with small basal leaves and large stem leaves as follows:

$$[0.5 \cdot (\overline{1}, \overline{1}), 0.5 \cdot (\overline{2}, \overline{2})]$$

4 GENERATING A DETERMINATION KEY

Determination keys are particular cases of *classification trees*. We review standard approaches to generate such trees from data.

4.1 Learning classification trees from examples

The problem of automating the reasoning of experts in a field (here: identifying a plant) dates back to the beginning of Artificial Intelligence, in particular with expert systems, where rules are created in collaboration with an expert of the field and a *knowledge engineer*. These rules tend to be deduction rules that are then fed to an inference engine that can use them to try and replicate predictions from expert. This process proved to be time-consuming and quickly people tried to have the computer learn the rules on its own. This gave rise to the field of *machine learning*, where the machine learns by itself, without the need of understanding how an expert reasons.

In the case of classification problems (such as determining a plant's species), the approach using classification trees as learnt knowledge became very popular. The computer learns from a set of examples a trees whose leaves contain a prediction for the category. Concretely an **example** is a tuple $(x_1, ..., x_n, y)$ where each $x_i \in X_i$ is an attribute value picked in a set of values X_i. X_i may be finite (for instance the petal colors) or infinite (length of the leaf). The final component, y is the class of the example, i.e. the expected result.

From such a set of examples, the goal is to learn classification tree: **leaves** are predictions, i.e. elements of Y, and **nodes** are attributes (e.g. x_2), with one child per value of the attribute.

The main metric for such trees is (1) *correctness* and (2) *conciseness*. One of the first system to learn such trees is CSL [Hunt et al. 1966], which uses a min-max approach to generate an efficient tree. This min-max approach involves exploring different branches of the tree to evaluate how optimal a condition is. This min-max approach tends to require a lot of computation, so other approaches were developed.

4.2 The ID3 algorithm [Quinlan 1986]

ID3 [Quinlan 1986] is a greedy algorithm that at every step picks the best condition according to some metric, and then recurses in the two sub-branch. Compared to CSL, there is no backtracking involved and so this algorithm is much faster. To get a performant tree, all the magic happens in the choice of the metric to evaluate the condition. In ID3, *entropy* is used to evaluate how well a condition partitions the set of possible predictions.

4.2.1 Data of the algorithm. The algorithm assumes a list of attributes, given as finite sets $X_1, ..., X_n$, and a set of Y. We assume a set $S \subseteq X_1 \times ... \times X_n \times Y$. The output of the algorithm is a tree, whose leaves are labelled with an element of Y, and nodes one of the X_i. A node labelled X_i has $|X_i|$ children.

4.2.2 Entropy. The entropy is a central notion in the ID3 algorithm which measures how uncertain a set of examples S is. The deeper we go in the tree, the smaller S becomes and thus the more certain it is.

Given a set S of examples and $y \in Y$, we define the probability of y relative to S:

$$p(y, S) := \frac{|S_y|}{|S|}$$

where S_y is the subset of S containing only examples with class y. The entropy of a set of examples is based on Shannon's formula [Shannon 1948] for entropy as follows:

$$H(S) := -\sum_{y \in Y} p(y, S) \times \log_2 p(y, S).$$

The minimal value of zero is reached when S only contains examples of the same class. In that case, the prediction is easy: it must be that class.

4.2.3 Description of the algorithm. The ID3 algorithm can thus be described as follows. It is parameterised by a set S of examples:

Algorithm ID3(S):

1. If $H(S) = 0$, emit a leaf labelled with the only class present in S.
2. Otherwise, for each attribute number $1 \leq i \leq n$:
   - Compute its score $C_i := \sum_{v \in X_i} p(v) H(S[x_i := v])$
3. Find the i with the lowest C_i.
4. For each $v \in X_i$ compute recursively $T_v := ID3(S[x_i := v])$.
5. Produce the tree with node labelled X_i and children $(T_v)_{v \in X_i}$.

where we use the following notations: $S[x_i := v]$ is the subset of S containing only examples where attribute i is v ; and $p(v)$ is defined as $|S[x_i := v]|/|S|$.

4.3 Adapting ID3 for our needs

There is a large body of work building on this algorithm trying to improve different aspects of it. One key aspect is avoiding overfitting, which is often done by pruning the tree after generation. This section is mostly informal, details can be found in Appendix A.

4.3.1 Bayesian approach. To take into account polymorphism, uncertainties and subjectivities our description are probabilistic. Thus, we need to turn the algorithm probabilistic, which means maintaining a distribution over species d rather than a subset $S \subseteq \mathcal{S}$ of possible species. Since the notion of entropy was defined first on probability distributions, this modifies little the algorithm. To compute the effect of an answer, we use Bayes' law (since our probabilities are discrete).

4.3.2 Structured description. Our description is **structured** instead of examples. To take this structure into account to improve the algorithm, we use a score function defined by induction:

$$\text{score} : \mathcal{O}(\mathbb{P}) \to \mathbb{O}(\mathbb{P}) \to [0, 1]$$

This function computes the conditional probability needed for the Bayes' law.

4.3.3 Discussion and shortcomings.

Prior distribution. One advantage of this approach compared to the one described in Section 4.2 is that the algorithm is parameterized by an initial probability distribution d_0 (the *prior* in bayesian term). This distribution can be taken to be the uniform distribution over the set of species $\mathcal{S}$. However, the weight of a species s can be related to its related to its frequency: the frequency in the region of observation is an a priori probability to observe a plant. In our proof of concept, we have used as a starting point, $n^{1/5}$ where n is the number of occurrences in the french territory as indicated by GBIF[7]. The low exponent is necessary to flatten the disparities between rarely and frequently observed and reported plants. Otherwise, rarely reported plants cannot be found with the key as their weight is too low.

This algorithm picks the most informative question, relative to the current probability distribution. In other words, it picks the question that is the best at discriminating the most likely species. This is an improvement over standard key algorithms that may pick questions to rule out very rare species.

Generating questions. In this algorithm, questions need to be specified by the creator of the key. From the description of the type $\mathbb{P}$, it is possible to generate a set of basic questions about one trait. For instance if $\mathbb{P}$ is described by:

```
plant := {
  leaf: [yes | no];
  flower_color: [ red | orange | yellow | blue ]
}.
```

Then we can easily generate two questions, one about the presence of the leaves and the second one about the color of flowers. In general we can generate a set of questions inductively on the structure of $\mathbb{P}$.

These questions are too sharp: for instance not everyone uses the same words to describe flower colors, and the boundaries between two colours, e.g. pink and red, are unsharp. To alleviate this, we equip every sum type appearing in $\mathbb{P}$ with a *confusion matrix* M that explicits how likely is it to confuse a trait value for another. For instance, here we could have $M_{\texttt{red,orange}} = 10\%$, while $M_{\texttt{red,blue}} = 5\%$.

Doing so protects us against user making mistakes while answering the question. Without the matrix, if the user answers `orange`, all the red flowers disappear from the distribution (their weight becomes zero). With the matrix, their relative weight is divided by five: they take a hit, but are still around. This mechanism can allow the user to make one or two mistakes in the exploration of the tree and still get a correct identification while standard determination keys tend to assume that the user is infallible.

Ambiguous questions. If $q = (o_1, ..., o_n)$ is a question and $d \in \mathscr{D}(\mathcal{S})$ a probability distribution, we can look at the sum:

$$\text{score}(o_1, d) + \cdots + \text{score}(o_n, d).$$

We would expect this sum to be equal to one, which means that every plant description is matched by exactly one of the o_i. This is not the case in general, especially for questions generated with a confusion matrix. We are interested in the case when the sum is greater than one, which means there is an overlap between different answers. This overlap may also come from the description of species present in d which may have several values for a trait.

This overlap quantifies how ambiguous the question is, and may reflect how likely a user may make a mistake while answering this question, *given the current distribution* d. While we have not tested it yet, we believe this quantity could be taken into account to make a determination key that instead of taking the most informative question, tries also to avoid too ambiguous question by picking questions for which the answer is more clear-cut from the current distribution.

Questions unlocking questions. When the sum is below one, the question is *partial*. This means there are individuals for which no answer really makes sense. This means typically that an answer is missing. This happens for generated questions for nested sub-types. Consider the following type for plants:

```
plant = { flower : [yes {color: [red | blue]} | no ] }.
```

This type has a single attribute, `flower` which can be `yes` or `no`. In the case of `yes`, we are also interested in the color of the petals. This type generates two questions: presence of flowers, and color of petals. It is obvious that the question about color is partial, indeed it only makes sense if we have established that the plant observed has flowers. Since we do not want to present a question to the user where they could get stuck, i.e. have no relevant answer, we need to ask questions in order. However, the question "is it a flowering plant" will often have a low score since most plants in identification keys are flowering. Thus the question will not get ask, even though it unlocks juicy questions such as the shape of the inflorescence that are highly discriminating. To circmvent this problem, we need to explore a few questions ahead to compute an amortized score taking into account questions that are unlocked by potential questions.

5 GATHERING AND USING DATA

To test the determination key generator we need a reasonable number of descriptions following our scheme. In this section, we describe our past and future attempts at gathering and formalizing morphological data. We have mostly considered three approaches: (1) mining existing sources online, (2) citizen science and (3) machine learning.

5.1 Database mining

So many databases of plant descriptions exist, either on books or digitised, that it is tempting to gather data from this previous work. However we were surprised by the recurrent obstacles. Firstly, very few databases have a permissive license, which is understandable because gathering data is a huge work and those who accomplished it fear that it is used without any benefit for them. We could not engage in individual collaborations with all of them, so we abandoned the idea of using a big part of the existing data. Secondly, datasets are not easily parsed. Let us take the example of the french flora by Bonnier [Gaston and de Layens 1909] which has been digitised by Tela-Botanica[8]. We attempted to translate the digitised questions and answers into our formalism. It is easy to obtain a graph whose nodes are questions (or taxons) and edges are answers to questions.

[7] Global Biodiversity Information Facility: https://www.gbif.org/fr/

[8] https://www.tela-botanica.org/eflore/bonnierpda/Bonnier.html

We had hoped at the beginning that we could just follow a path to a species and get partial data for that species as the conjunction of its edges. This proved difficult because, we had expected the graph to be a tree (as in *decision tree*) but it is actually a DAG, that is there are several paths leading to the same species. This phenomenon is due to the key having a particular feature: it first identifies the family, so several paths converging to a single family do not necessarily concern the identified species in question.

We have also tried to parse the data from TRY, which is a trait database for plants. Although it contains a lot of data, it actually is a "database of databases", thus including many sources which are sometimes contradictory, and contains a lot of missing data. As explained above, it was mainly conceived for plant ecological studies with the goal to compare plants, not to distinguish them, and thus not contain a lot of useful traits for plant identification.

5.2 Manual description filling

So we explored ways to reconstruct from scratch the datasets, informed by existing floras, digitised or not, but manually filling the fields instead of automatically mining. We have designed an interface, parameterized by the `plant.scheme` that allows to input the description of a species. Concretely, it means for each sum type in the scheme, giving the value(s) for that particular species. Since our model is probabilistic, we can also specify a probability distribution on values for a trait, if there are several. The editor takes advantage of the hierarchy of the `plant.scheme`: when selecting a trait value, this may unlock new traits that were depending on it. For instance, in the case of inflorescence, if we select `capitulum`, this would unlock the traits relative to them. This allows us to only see the traits that are relevant to the current species being edited, and also to take it slowly, as the traits unfold the more we add information.

Data for a species is usually filled from some bibliographical source. Due to the polymorphism and the uncertainty evoked above, there is in general not a complete consensus about the existing plant observation in the literature. For the database to be scientific value, it is important to be able to source each value in the entry for a species (especially the least obvious ones). To that end, the editor allows to add bibliographic sources for each entry.

5.2.1 Experiments in citizen science. We would also like to involve associations interested in conservation and environment education to help us fill the database. We have obtained a small grant to work with local and regional associations. The goal is to use the knowledge from the employees and volunteers of the association in exchange for producing determination keys for areas of interest. This is also a good way for people to strengthen their botanical knowledge, which is often a side goal of these associations. We have devised the following methodology so far, where the goal is to study a small ecosystem. Our partnership works as follows:

1. **First fieldwork:** Botanical inventory of the ecosystem, paying attention to how frequent the species are (this is needed to get a good prior distribution).
2. **Lab session:** Adding missing species to the database.
3. **Second fieldwork:** After generating the key, we try to validate it going back to where the inventory was made.

This methodology strengthens also a goal of the project which is to get interested citizens more involved with their local ecosystems. For a given environment, the methodology can be repeated several times across the year as the species distribution tend to vary a lot. This opens the door of making seasonal determination keys that could even be more effective. Indeed, the current season tends to be the first criteria used by botanists to guide what species they are expected to find (and in what form).

5.2.2 Partnering with national networks of botanists. We also are in the process of partnering with BotaScopia[9] a french national initiative led by Tela-Botanica and Université Paris-Saclay aiming at providing uniform descriptions of species based on a formal representation. In this project, the data is entered by students for which this process is part of their curriculum.

5.3 Machine Learning

Another way to obtain data is to use machine learning techniques. We see two possibilities.

5.3.1 Using NLP. There is a way to use NLP techniques to retrieve information about natural language sources such as floras. It is important that the model is able to source its values for attributes. We have done some experiments with ChatGPT but they still have to be thoroughly investigated. Before, however, we should ask ourselves whether using such a tool, given its energy consumption and its use of "human resources" in training it fits with our general objectives. It seems preferable to directly work with botanists as entering botanical data is meaningful for them.

5.3.2 Using image recognition. Deep learning could be imagined to associate photos with traits, based on a training set. In this way a subset of traits might be immediately recovered from a simple image. However this might not be compatible with our idea of staying in certain limits in terms of computations and resource usage.

6 CONCLUSION AND FUTURE WORK

Determination keys are algorithms. They consist in deterministic sequences of operations that lead to identification. This makes appealing the idea to use some computer science theory to analyse, generate and improve them. Surprisingly, even if computers have been used a lot to help the botanical work, close concepts from theoretical computer science (bayesian inference, decision trees, information theory, formal languages) have been little explored in this scope, apart from deep learning and notable exceptions like xper [Kerner and Lebbe 2019].

We have presented a methodology for formalising botanical knowledge, generating data according to this formalisation, and using the data to generate determination keys. We believe that this can be an improvement over the existing techniques for handling in a probabilistic framework polymorphism, errors, uncertainties, and proposing a hierarchical evolvable language that may solve the conundrum between precision and accessibility of plant descriptions. The system presented here is very much a work in progress. Some aspects have been implemented, and we are running experiments.

Ultimately, we would like the determination keys to be possibly printed on paper, at least for small floras, as well as being usable

[9] https://www.tela-botanica.org/2022/05/bientot-un-outil-de-creation-de-fiches-en-botanique/

offline on a modest smartphone or computer. Instead of a giant national or international flora, we would also like to favour smaller floras more adapted to specific ecosystems. In this way the computing systems are used to construct objects, and these objects, and not the computers, are the daily used objects.

In this sense this work is an occasion to explore concretely innovative computer science concepts such as *self-obviating systems* and *transition computing*, related to *conviviality*.

It is necessary to discuss how to evaluate a priori and a posteriori the claim of convivial and self-obviating characters of the tools we propose. According to Illich [Illich 1973], "The simple, poor, transparent tool is a humble servant; the elaborate, complex, secret tool is an arrogant master." We will not achieve conviviality when we rely on computer tools connected to the Internet that require a complex arrangement of technologies. However compared to AI systems we can argue that we progress in the three cited dimensions: compared to AI based identification tools, our planned tools are simpler. Not in their usage but in their conception. Indeed our proposed techniques require some amount of computer science development and botany knowledge, however these can be described without the reference to theories that require a deep involvement to understand as neural networks. Furthermore, our tools can be considered a progress on the transparency vs secrecy dimension. Indeed, current AI tools, even if explainability is an active field of research, never reveal their tricks, even though they are necessarily based (implcitly or explicitly) on already available human knowledge. Finally we will have progressed on the poor vs complex dimension if for small floras we can generate and print keys on papers. Then we will have achieved the poor, transparent and humble qualities.

The explainability property is also a wished path to self-obviating systems. Every use of the application should make it less useful for a user, because every answer of the algorithm goes with a careful observation of a plant and with an association of the observation and the result. This remains to be evaluated: a posteriori, it will be possible to measure if the users of explainable tools feel more autonomous in their dialog with their vegetal environment than AI users. Note that self-obviating for a user does not mean self-obviating for a society: there will always be users who need training, so that the tool never self-disappears eventually. However given that we are living an important transformation of our societies due to social and environmental crises, it is reasonable to consider all tools as transitory, and their effects in this transition more than in a stable society.

Eventually we claim the development of an *humble computer science*. Humble in several meanings: (1) we use much less resources than traditional AI approaches (2) we build on, and with the people who already have the botanical knowledge and (3) we consider computer science knowledge as a "back-office" supporting external research questions, more than an "avant-garde" that would transform the society by itself, to borrow the words of Bruno Latour during his very last conference in Paris [Latour 2022]. Our goal is not to replace botanists but to spread their knowledge and make it accessible.

We believe our framework will be helpful for training botanists, both formally (biology students) and informally (gardeners, amateurs). Novices in botany could use a key based on the common plants in their region, restricted to the species of interest (they often don't start by identifying grasses), which will generate a rather easy-to-use key. If the users progress, they might want to include rarer species, or those occurring in a larger region, or those that are notoriously difficult to identify. The keys will become increasingly precise, requiring more botanical knowledge, but this procedure offers the possibility to learn that knowledge step-by-step. Such an incremental learning procedure is not possible with classical floras and thus could have many applications in teaching and participative science.

Finally, this work can have benefits outside the generation of determination keys. Formalized botanical knowledge is useful in evolutionary biology, ecology, and conservation biology. The question of why there are so many plant species, and why some groups are more species-rich than others (e.g. flowering plants vs. gymnosperms) is still largely unanswered. Morphology is at the intersection of evolution and ecology, as some traits are clearly adaptations to enhance survival in certain environments (e.g. thorns prevent plants to be eaten by large herbivores, leaves densely covered with hairs protect against sunlight and evaporation), while others are the result of evolutionary and developmental constraints (e.g. parallel leaf nerves probably don't influence the species' ecological interactions). Among others, the morphology of the species can be used to quantify structural and functional diversity, allowing to better understand the functioning of ecosystems than by species numbers alone. Morphological descriptions should be available for all species that have been described so far, thus allowing biodiversity studies to benefit from centuries of botanical descriptions without the need to spend time, human work and physical energy, all of which are limited, for fieldwork to collect the data.

REFERENCES

2012. OWL: Web Ontology Language. https://www.w3.org/OWL/.

Antoine Affouard, Jean-Christophe Lombardo, Hervé Goëau, Pierre Bonnet, and Alexis Joly. 2019. Pl@ntNet. https://hal.archives-ouvertes.fr/hal-02096020

Jean-Baptiste de Lamarck. 1779. *Flore française.*

Bonnier Gaston and Georges de Layens. 1909. *Flore Complète De La Suisse Et De La France Pour Trouver Facilement Les Noms Des Plantes Sans Mots Techniques.*

Jean-Yves Girard. 1987. Linear logic. *Theoretical computer science* 50, 1 (1987), 1–101.

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. 2018. The INaturalist Species Classification and Detection Dataset.. In *CVPR*. IEEE Computer Society, 8769–8778. http://dblp.uni-trier.de/db/conf/cvpr/cvpr2018.html#HornASCSSAPB18

Earl B Hunt, Janet Marin, and Philip J Stone. 1966. Experiments in induction. (1966).

Ivan Illich. 1973. *Tools for conviviality.*

Pankaj Jaiswal, Shulamit Avraham, Katica Ilic, Elizabeth A Kellogg, Susan McCouch, Anuradha Pujar, Leonore Reiser, Seung Y Rhee, Martin M Sachs, Mary Schaeffer, et al. 2005. Plant Ontology (PO): a controlled vocabulary of plant structures and growth stages. *Comparative and functional genomics* 6, 7-8 (2005), 388–397.

Adeline Kerner and Régine Vignes Lebbe. 2019. Multi-context Knowledge Base using Calculated Descriptors from Xper3: the Archaeocyaths Knowledge Base example. *Biodiversity Information Science and Standards* 3 (June 2019). https://doi.org/10.3897/biss.3.37083

Bruno Latour. 2022. The new free university. (2022). https://www.sciencespo.fr/com/news/discours-B.Latour-FR.pdf

Robin Milner, Mads Tofte, Robert Harper, and David MacQueen. 1997. *The definition of standard ML: revised.* MIT press.

J. Ross Quinlan. 1986. Induction of decision trees. *Machine learning* 1 (1986), 81–106.

Hervé Sauquet, Maria von Balthazar, Susana Magallón, James A. Doyle, Peter K. Endress, Emily J. Bailes, Erica Barroso de Morais, Kester Bull-Hereñu, Laetitia Carrive, Marion Chartier, Guillaume Chomicki, Mario Coiro, Raphaël Cornette, Juliana H. L. El Ottra, Cyril Epicoco, Charles S. P. Foster, Florian Jabbour, Agathe Haevermans, Thomas Haevermans, Rebeca Hernández, Stefan A. Little, Stefan Löfstrand, Javier A. Luna, Julien Massoni, Sophie Nadot, Susanne Pamperl, Charlotte Prieu, Elisabeth Reyes, Patrícia dos Santos, Kristel M. Schoonderwoerd, Susanne Sontag, Anaëlle

Soulebeau, Yannick Staedler, Georg F. Tschan, Amy Wing-Sze Leung, and Jürg Schönenberger. 2017. The ancestral flower of angiosperms and its early diversification. *Nature Communications* 8, 1 (Aug. 2017). https://doi.org/10.1038/ncomms16047

Claude E Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal* 27, 3 (1948), 379–423.

Société botanique de France. 2014. *Flora Gallica: Flore de France.* Biotope Editions.

Bill Tomlinson, Juliet Norton, Eric Ps Baumer, Marcel Pufal, and Barath Raghavan. 2015. Self-Obviating Systems and their Application to Sustainability.

T G Tutin, V H Heywood, N A Burges, D H Valentine, S M Walters, and D A Webb. 2001. *Flora Europaea 5 volume set and CD-ROM pack.* Cambridge University Press, Cambridge, England.

A OUR BAYESIAN ALGORITHM

We present here our variation on ID3 that aims at addressing some of the issues presented in the previous section. It is nowhere near perfect and there are many choices still to be discussed. This is a part of the design space that we have not yet been able to explore in depth, in particular because we had to focus on building the database first.

A.1 Questions and observations

As we have seen, the ID3 algorithm works as input with a set of observations $(x_1, ..., x_n, y)$. Another way to see it is that each y has a set of observations. For traditional applications of ID3, these observations are not necessarily exhaustive, thus the danger of over-fitting.

In our setting, we fix a set $\mathcal{S}$ of species along with a map $\delta : Y \to \mathcal{O}(\mathbb{P})$ mapping each species to a probabilistic description of it. The pair $(\mathcal{S}, \delta)$ is the database.

In ID3, the questions of the decision tree are given by the attributes. We choose here thus to derive the questions from the structured description $\mathbb{P}$.

A question is a list of **observations** $(o_1, ..., o_n)$ where $o_i \in \mathcal{O}(\mathbb{P})$. Each o_i represents a possible answer to the question. We assume fixed a set of questions $\mathcal{Q}$. We are going to apply the ID3 algorithm where attributes are questions and attribute values are answers.

A.2 Definition of the score function

Given a question $q = (o_1, ..., o_n)$, and a class $s \in \mathcal{S}$, we need to compute which answer is relevant for it. Because we are in a probabilistic setting, for each answer o_i, we want to compute the likelihood of s satisfying answer o_i. This weight is defined by a score function $\text{score}(o, d)$ where the first argument is the observation (answer) and the second is the description of the species. It returns a number which says how likely an individual of species described by d matches o. The function is defined by induction on $\mathcal{O}(\mathbb{P})$, which guarantees that the two observations have the same structure.

Scores for record and sum types are just taken from the formula for conjunction and (probabilistic disjunction):

$$\begin{aligned} \text{score}(\langle o_1, ..., o_n\rangle, \langle d_1, \ldots, d_n\rangle) &= \textstyle\prod_{i\in[1,n]} \text{score}(o_i, d_i) \\ \text{score}(\oplus_{i\in I} w_i \cdot o_i, \oplus_{i\in I} v_i \cdot d_i) &= \textstyle\sum_{i\in[1,n]} w_i \times v_i \times \text{score}(o_i, d_i) \end{aligned}$$

The score formula for multiple types is a bit more subtle:

$$\begin{aligned} &\text{score}([p_1 \cdot o_1, ..., p_i \cdot o_i], [q_i \cdot d_1, ..., q_j \cdot d_j]) \\ = \ &\textstyle\prod_{i\in I} p_i \times \sum_{j\in J} q_j \times \text{score}(o_i, d_j) \end{aligned}$$

The choice on the left is understood as an implicit conjunction ("we have observed leaves like this and like that"), while on the right it is an implicit disjunction ("species may have leaves like this or like that"), hence the product of sums.

Finally, for the case of quantitative trait, it depends on how they are represented concretely. Our implementation has only normal distributions (μ, σ), and thus the score can be implemented as follows:

$$\text{score}((\mu_1, \sigma_1), (\mu_2, \sigma_2)) = d_{(\mu_2,\sigma_2)}(\mu_1)$$

where $d_{(\mu_2,\sigma_2)}$ is the density function of the normal distribution with parameters μ_2, σ_2.

A.2.1 Bayesian aspect. Another aspect we need to improve on the standard formulation of the ID3 algorithm is to use a bayesian approach. Concretely, instead of maintaining a set S of candidates, refined as we walk through the trees, we keep a probability distribution $d \in \mathcal{D}(\mathcal{S})$. Since the concept of entropy was originally formulated at the level of distribution this part is easy to extend:

$$H(d) = -\sum_{s\in\mathcal{S}} d(s) \times \log_2 d(s).$$

The main difficulty is to extend the construction $S[x_i := v]$ which restricts a set of candidates to those for which $x_i := v$. In this setting, we define the operation $d[o]$ which updates a probability distribution with the knowledge of observation o using Bayesian inference. As noticed above, $\text{score}(o, s)$ corresponds to the conditional probability of observing o, *knowing* we are observing s. This means that we can define $d[o]$ as follows from Bayes' law:

$$d[o](s) = d(s) \times \frac{\text{score}(o, s)}{\sum_{s'\in\mathcal{S}} \text{score}(o, s')}$$

The denominator is simply a normalising factor.

Using those two notions, we can define the entropy of a question as the average of the entropy of its answers. Given $d \in \mathcal{D}(\mathcal{S})$ and $q = (o_1, ..., o_n)$ we let

$$H(d, q) := \sum_{1\le i\le n} \text{score}(o_i, d) \times H(d[o_i])$$

Using this, we can define the score function to a probability distribution of species, instead of a single species observation. Given a observation o in a probability distribution $d \in \mathcal{D}(\mathcal{S})$, we let:

$$\text{score}(o_i, d) := \sum_{s\in\mathcal{S}} d(s) \times \text{score}(o_i, \delta(s))$$

A.3 Final algorithm

Using the previous definitions, we can now define our algorithm:

Inputs: A database $(\mathcal{S}, \delta)$; a set of questions $\mathcal{Q}$.

Outputs: A tree, whose nodes are labelled with elements of $\mathcal{Q}$, and leaves with distributions over $\mathcal{S}$. Each node labelled q has a child per answer of q.

The recursive algorithm is parameterized by a distribution $d \in \mathcal{D}(\mathcal{S})$ and $\mathcal{Q}_0$ the remaining questions.

Algorithm ID3'($\mathcal{Q}_0$, d):

1. If $\mathcal{Q}_0$ is empty, return a leaf labelled with d.
2. If d is a Dirac distribution, return a leaf labelled with d
3. Find the question $q = (o_1, ..., o_n) \in \mathcal{Q}$ that maximises $H(d, q)$.

4. Return the tree with root node labelled q and children, the family:

$$(\mathrm{ID3}'(\mathcal{Q}_0 \setminus \{q\}, d[o_i]))_{1 \leq i \leq n}.$$