Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Master's Thesis Nr. 102

Systems Group, Department of Computer Science, ETH Zurich

Expert System for Identification of Trees

by

Simon A. Eugster

Supervised by

Prof. Donald Kossmann

April 15th to October 15th 2013

Informatik
Computer Science

Contents

1 Introduction

Summary This thesis examines current identification keys and their advantages and issues, and describes the design of a generic web-based identification key with those issues addressed.

Terminology The term *taxon* is used for a group of items which share the same name, like trees *(Fraxinus excelsior),* clouds *(Cumulus),* or light bulbs *(halogen lamp).* The term *character* is used for a feature that can be used to distinguish between taxa. The term *key* is sometimes used as the short form for *identification key.*

1.1 Background and Motivation

This section briefly describes some aspects that can be improved in current identification keys.

Keys are used to find out what something is, to **find the name of a thing.** Why is that important at all?

For me, it is often curiousity. But there are good reasons as well; figure 1.1 mentions one. If the sky is covered with thick *Altostratus* clouds, I know that it will most likely be raining for a longer time in a few hours. Unlike so with *Stratus*, a similar-looking cloud, which never produces rain. Or, if I need wood for making bows, I have to know if this tree is an European Ash *(Fraxinus excelsior)* or a Beech *(Fagus sylvatica):* the former is great bow wood, whereas the latter usually breaks in heavier bows.

Figure 1.1: Is this mushroom edible? Identification provides an easy way for finding out without dangerous tasting.

But there are other motivations as well. A prominent one is the desire for knowing all life on Earth; that in order to capture biodiversity and as *a starting point for halting biodiversity decline*, as stated in LaSalle et al. (2009). The authors estimate the discovery and description of all species—which again are estimated to around 10 million—to several centuries when proceeding at the same speed.

Traditional keys are in the form of books. Switzerland has its keys for the local species (often in two or three languages), Germany has them (in German), France (in French), England (in English), Mexico (in Spanish), and so on. Perhaps any country can be named.

Already three problems arise from this form. First, if I want to identify a tree in Mexico, I may have to learn Spanish first; the Swiss keys do not describe the trees growing there. Second, an identification key to all trees would result in a multi-volume and little portable book. Third, the information is printed and therefore static; the key cannot be adjusted according to decisions made by the user, and new taxa could only be added in a new edition.

Another point is a bit more subtle, but just as important: finding a suitable key at the first place. Many are, as mentioned, in the form of books. Others, especially specialised ones, can only be found in papers and have never been printed.

All those **problems can be solved with a digital identification key.** It can be translated, has much less storage issues, and is dynamic. As a bonus, digital keys can be made freely accessible and editable—Wikipedia and its sister projects serve as great examples there.

With more storage available, the keys may also contain more sample images. Identification keys in books are often illustrated very sparsely.

1.2 Problem Statement

This section lists the key features which the identification key written in this thesis aims to fulfil.

The main reason for me to start this thesis was the observation that dichotomous keys in books are difficult to use as soon as a character required for the next step cannot be determined. (This problem will be explained more in-depth in section 2.1.) The previously mentioned observations led to additional points that shall be addressed.

My vision is an open computer-based identification system, accessible to and editable by everyone. It should be general enough to support various types of subjects besides trees, which serve as example in this thesis.

Editability The key should be editable, in a way simple enough that also "normal" people—i.e. not only computer experts—can add taxa and identification criteria. (Biologists, for example, usually are not computer experts.) Many projects prove that collaboration is the only way to keep them alive; closed-source projects with sole authors are usually attended few years only and then forgotten. Collaboration is, at the same time, a great way to gather information from all over the world in a central, open resource—just like Wikipedia does it, and very sucessfully so.

Performance In terms of both speed and usability. Performance can be defined by the cost of identifying species, like the number of questions asked or the "sum" of their difficulty, which is to be minimized. Small differences thereof may not be noticeable, however pointless questions that e.g. don't even narrow down the remaining selection are. Performance can also be defined in terms of usability: The user interface has to respond quickly and avoid useless interactions; an example of latter would be a confirmation dialog *"are you sure?"* whenever a character is chosen.

Internationality To make a key usable globally, such that e.g. also Asian tourists can identify plants growing on the Swiss mountains, it has to be translated. The same holds if editors of different countries work on the data. The taxon characters themselves—their meaning—do not differ, no matter what language one speaks, only their localised description does. Consequently, ideally only the character description is translated and the features are defined only once, "for the whole world".

Growth Every project needs to reach a "critical mass" until it starts growing by itself. For my thesis, this affects both code and data. Enough data is necessary so contributors do not have the feeling of contributing to an empty project; I regard a sample key as mandatory, and its data can additionally serve as real-life test set. The code—especially the identification part—has to be good enough to make people want the program to identify their trees as well, and well designed—i.e. not hacked together—so other programmers can work on it without too big effort.

1.3 Structure of This Thesis

Chapter 2 gives an overview over the identification keys that are currently used—key systems as well as concrete examples of them—and examines their advantages and disadvantages. Knowledge about those is mandatory for understanding the further decisions and reasonings. Chapter 3 contains the technical part of this thesis; it describes the identification key system I developed. It should be easy to read also if you are not a computer scientist. Chapter 4 finally discusses the results and possible future work.

Each section starts with a short *summary in italic*.

2 State of the Art

Allow me to start this chapter with a 33 years old quote:

Identification keys and diagnostic tables are simple to use and easy to carry, so we think they will retain their popularity until we all have our own pocket micro-processor. —Payne and Preece, 1980

2.1 Key Systems

This section gives an overview over existing types of keys and discusses advantages and disadvantages of each.

2.1.1 List of Images

Image lists are easy to read and ideal for small data sets.

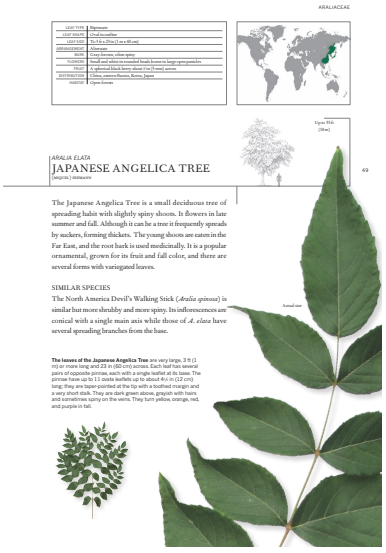
The most common kind of key encountered is the *field guide*. A vast number of books on any subject can be found: plants, birds, ants, fishes, constellations, insects, and so on. Those keys need no introduction how they work; **comparing images with the object to identify is a task our brain is excellent at.** For example, consider how quickly you can identify a person you know on the street, although there are many other faces around. This is *amazing*.

(This may sound like something obvious, but consider how hard it still is for a computer to accomplish such tasks, whileas it is millions of times faster in multiplying matrices than I am.)

Generating such keys is also relatively easy—which does not mean that it is little work!—, not much more but an image or a painting and the name is required.

So, why is this not enough? Those keys become more inefficient the larger the data set is, it is arduous to match dozens and hundreds of pictures against the object, and similar looking species—thinking of living creatures—make the matching process error-prone.

Technically speaking, searching takes $\mathcal{O}(n)$ time. Extended versions coarsely groupo taxa e.g. by flower colour or other characters to speed up the scanning process.


ARALIACEAE

ARALIA ELATA

JAPANESE ANGELICA TREE

49

The Japanese Angelica Tree is a small deciduous tree of spreading habit with slightly spiny shoots. It flowers in late summer and fall. Although it can be a tree it frequently spreads by suckers, forming thickets. The young shoots are eaten in the Far East, and the root bark is used medicinally. It is a popular ornamental, grown for its fruit and fall color, and there are several forms with variegated leaves.

SIMILAR SPECIES

The North America Devil's Walking Stick (*Aralia spinosa*) is similar but more shrubby and more spiny. Its inflorescences are conical with a single main axis while those of *A. elata* have several spreading branches from the base.

The leaves of the Japanese Angelica Tree are very large, 3 ft (1 m) or more long and 23 in (60 cm) across. Each leaf has several pairs of opposite pinnae, each with a single leaflet at its base. The pinnae have up to 11 ovate leaflets up to about 4 in (12 cm) long; they are taper-pointed at the tip with a toothed margin and a very short stalk. They are dark green above, grayish with hairs, and sometimes spiny on the veins. They turn yellow, orange, red, and purple in fall.

Figure 2.1: *The Book Of Leaves* (Coombes, 2011) contains leaf images of 600 trees. It is amazing how quickly trees can be found merely by looking through the 30 pages with preview images in the "key" section.

2.1.2 Dichotomous Key

Dichotomous keys are binary search trees and ideal for large data sets.

An advanced method is the dichotomous key, which is a character based binary search tree. Each node represents a character, and its subtrees are chosen depending on whether the character matches or not. The leaves are, for example, families or species. Their big advantage is that they can **cover much larger numbers of species** since the key's depth is only $\mathcal{O}(\log_2 n)$ in case it is balanced. It also **forces the user to look more carefully** at the examined object in order to correctly identify the required characters, and hereby teaches to—and how to—see things invisible to the untrained user.

Dichotomous keys have disadvantages too. Many of them are merely text, without images, which makes them look too dry or complicated especially for occasional users. Some characters require specialist knowledge that is not always explained along with the key, or they require special tools like magnifying glasses or microscopes. For those reasons, dichotomous keys are mainly used by experts.

The structure of dichotomous keys leads to another problem: If a character cannot be identified definitely, both subtrees need to be followed, requiring the user to jump forth and back between multiple options and check which suit better. If a character is incorrectly identified, one may have to re-start close to the root and re-check all decisions, or it may even be impossible to correctly identify the taxon.

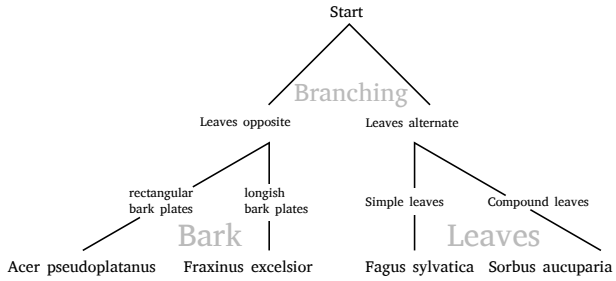


Figure 2.2: Simple dichotomous key for four taxa.

Finally, although the tree depth is only $\mathcal{O}(\log_2 n)$ compared to $\mathcal{O}(n)$ for lists, it takes longer for small collections of, for example, 20 species, since the eye is much faster matching them visually than identifying characters and reading text.

The following improvements can be made to dichotomous keys:

Illustrations Adding illustrations of the characters (figure 2.3) speeds up identification, as explained in the previous section about lists. For the same reason—images are natural to our brain, text is not—illustrations also let the key look much "lighter" and easier to understand.

Multichotomous key To reduce the depth of the tree, and hereby the number of questions asked, multiple options for a character can be given at once. For example, for species of the familiy *Pinus* one wants to know whether they have 2, 3, or 5 needles per "bundle" (fascicle). A strict BST (binary search tree) would require two questions: *a) Are there 2 or more needles per fascicle?* and *b) Are there 3 or 5 needles?*

Taxon descriptions Description of additional *redundant* characters for the found taxon help in ensuring that the identification was actually correct. For the example key shown in figure 2.2, the tree bark may be hard to classify, or may not even show the characters yet if it is young. Providing additional information for the Ash *(Fraxinus excelsior)* like "buds are black" then makes confusion with *Acer pseudoplatanus* impossible.

2.1.3 Diagnostic Tables

Diagnostic tables are matrices with the taxa on one axis and the characters on the other. They provide most information, but are slowest to search manually.

The most general form of a key is the diagnostic table. One axis of the table contains the taxa, the other one the characters; thus they **contain most information on characters** of the keys listed here. Searching diagnostic tables by hand is perhaps nearly as fast as for a dichotomous key for small data sets. When it grows, sorting is essential; the example table below is first sorted by *Branching* and then by *Leaf Type*:

	F. excelsior	*A. p.platanus*	*S. aucuparia*	*F. sylvatica*
Branching	opposite	opposite	alternating	alternating
Leaf Type	compound	simple	compound	simple
Bud Color	black	green	grey	reddish
Leaflets	9–15	—	9–19	—

If the first column's character cannot be identified, the user has to jump around in the table; the same problem known from dichotomous keys. This

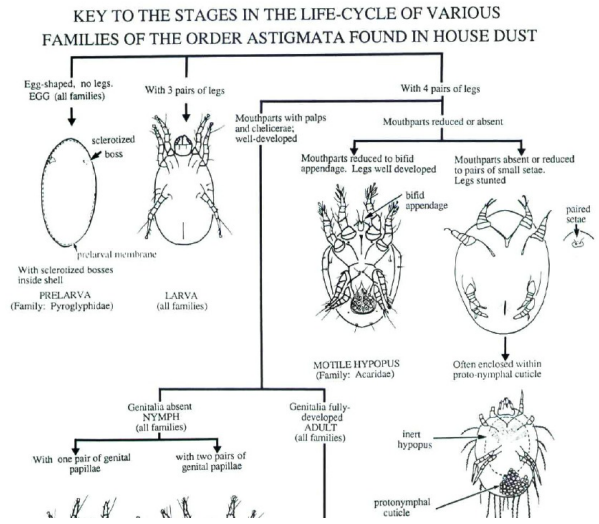


Figure 2.3: Identification key for domestic mites, with illustrated characters, from Colloff and Spieksma (1992)

	Arginine dihydrolase (Møller)	Lysine decarboxylase (Møller)	Ornithine decarboxylase (Møller)	Growth at 0% NaCl	Growth at 3% NaCl	Growth at 6% NaCl	Growth at 8% NaCl	Growth at 10% NaCl	Growth at 4°C	Growth at 20°C	Growth at 30°C	Growth at 35°C	Growth at 40°C	Aesculin hydrolysis	Amygdalin	Citrate	Gelatinase	Glucose gas
Phenon 10/85	–	–	–	+	ND	(–)	ND	ND	–	ND	ND	ND	ND	v	–	+	–	–
Phenon 14	+	–	–	–	ND	–	ND	ND	–	ND	ND	ND	ND	v	–	+	+	–
Phenon 2	–	–	–	–	ND	v	ND	ND	–	ND	ND	ND	ND	v	–	+	+	–
Phenon 21	–	–	–	–	ND	–	ND	ND	–	ND	ND	ND	ND	v	–	(–)	+	–
Phenon 5	v	–	–	–	ND	v	ND	ND	v	ND	ND	ND	ND	+	v	+	–	v
Photobacterium angustum	+	v	–	–	+	+	–	–	(+)	+	+	+	–	v	–	–	(+)	–
Photobacterium leiognathi	(+)	v	–	–	+	+	–	–	–	+	+	+	–	v	–	–	–	(–)
Photobacterium phosphoreum	+	(+)	–	–	+	v	–	–	+	+	(+)	–	–	v	–	–	–	(+)
Plesiomonas shigelloides	+	+	+	+	+	–	–	–	+	+	+	+	–	–	–	v	–	(–)
Vibrio aestuarianus	+	v	–	–	+	+	v	–	+	+	+	+	–	ND	+	+	+	–
V. alginolyticus	–	+	v	–	+	+	+	+	–	+	+	+	+	(–)	–	+	+	–
V. anguillarum	+	–	–	(–)	+	d	–	–	d	+	+	+	–	(–)	–	+	+	–
V. anguillarum-like	+	v	–	–	ND	–	ND	ND	+	ND	ND	ND	ND	–	–	+	+	–
V. campbellii	–	d	–	–	+	d	v	–	–	+	+	+	–	v	–	v	+	–
V. carchariae	–	+	+	–	+	+	+	–	–	+	+	+	+	ND	ND	–	+	–
V. cincinnatiensis	–	(+)	–	–	+	+	+	v	–	ND	+	+	ND	ND	–	+	–	–
V. costicola	(+)	–	–	–	+	+	+	(+)	d	+	+	+	–	v	–	–	–	–
V. cholerae	–	+	+	+	+	–	–	–	–	+	+	+	+	–	–	v	+	–
V. damsela	+	v	–	–	+	+	v	–	–	ND	ND	ND	ND	–	–	–	–	d
V. diazotrophicus	(+)	–	–	–	+	+	+	v	(+)	ND	+	+	v	+	–	+	–	–
V. fischeri	–	+	(–)	–	+	d	–	–	(–)	+	+	v	–	–	–	v	–	–
V. fluvialis	+	–	–	v	+	+	(+)	(+)	–	+	+	+	+	v	(+)	+	+	–
V. furnissii	+	–	–	d	+	+	+	+	–	+	+	+	+	–	–	+	(+)	+
V. gazogenes	–	–	–	–	+	+	+	+	–	+	+	+	+	ND	ND	+	+	+
V. harveyi	–	+	+	–	+	(+)	(+)	(–)	–	+	+	+	v	(–)	–	v	+	–
V. hollisae	–	–	–	–	+	+	+	(–)	ND	ND	+	+	ND	–	ND	–	–	–
V. logei	–	+	(+)	–	+	–	–	–	+	+	–	–	–	ND	ND	–	–	–
V. marinus	–	+	–	–	+	–	–	–	+	+	–	–	–	ND	ND	–	+	–
V. mediterranei	d	v	–	–	+	+	v	–	–	+	+	(+)	–	+	ND	–	–	–
V. metschnikovii	d	v	–	d	+	(+)	(+)	(–)	d	+	+	+	+	v	–	v	+	–
V. mimicus	d	+	+	+	+	d	–	–	–	ND	+	+	v	–	–	+	+	–
V. natriegens	–	–	–	–	+	d	+	–	–	+	+	+	+	+	v	+	(+)	–
V. nereis	+	–	–	–	+	d	+	(–)	(–)	+	+	+	v	(–)	–	+	v	–
V. nigripulchritudo	–	–	–	–	+	d	–	–	–	+	+	–	–	v	–	+	+	–
V. ordalii	–	–	–	–	+	(+)	+	–	d	+	+	–	–	v	–	+	+	–
V. orientalis	d	+	–	–	+	+	+	–	+	+	+	+	–	ND	ND	+	+	–
V. parahaemolyticus	–	+	+	–	+	+	+	–	–	+	+	+	+	–	–	+	+	–
V. pelagius I	–	–	–	–	+	+	+	–	v	+	+	+	–	ND	ND	+	–	–
V. pelagius II	–	–	–	–	+	+	+	–	v	+	+	+	–	ND	ND	+	+	–
V. proteolyticus	+	+	–	–	+	+	+	+	–	+	+	+	+	–	–	+	(+)	–
V. salmonicida	–	–	–	–	v	–	–	–	+	+	–	–	–	–	–	–	–	–
V. splendidus I	(+)	–	–	–	ND	d	v	–	d	+	+	v	–	v	–	d	+	–
V. splendidus II	v	–	–	–	ND	d	v	–	–	+	+	–	–	v	–	d	+	–
V. tubashi	(+)	–	–	–	ND	d	v	–	–	ND	v	v	–	–	–	+	+	–
V. vulnificus	–	+	+	–	+	d	–	–	–	+	+	+	+	(–)	–	+	+	–
V. vulnificus B2	–	+	–	–	+	+	–	–	–	+	+	+	ND	ND	+	+	+	–

ND, No data; d, discrepancies (see text); +, positive for ≥90%; (+), positive for 75–89%; –, negative for ≤10%; (–), negative for 25–11%; v, variable for 26–74%. Percentages indicate positive results.

Figure 2.4: Diagnostic table for Vibrio species (Alsina and Blanch, 1994)

has early been worked around with software; (Payne and Preece, 1980, Section 4.5) mentions systems from the sixties. In software it is trivial to filter the table by the given criteria.

As the table grows, it is easy to lose overview. Already the diagnostic table in figure 2.4 is not easy to decipher as the user's eyes have to keep track of both horizontal and vertical position, and it has no more than 18 characters.

Yet, diagnostic tables are *very* easy to evaluate with software. They will therefore also be the key system of choice for this thesis.

2.2 Examples of Existing Keys

This section lists some existing keys on different media and discusses advantages and disadvantages, helping to clarify the project goals.

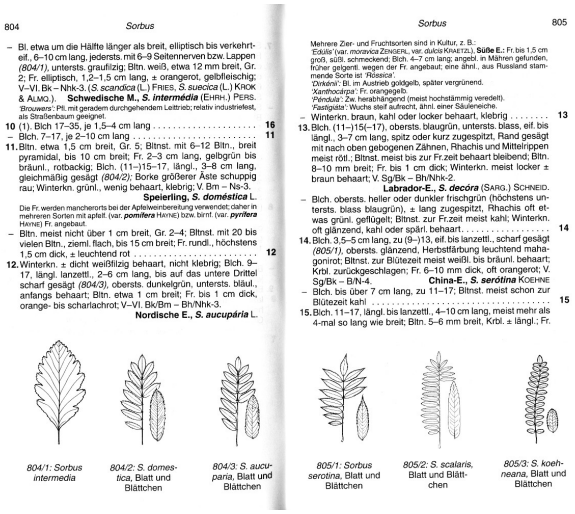
It is amazing to see how many keys exist. The web makes it even easier to find them, from all over the world. From general plant keys to very specific ones for families, from mites to fungi, for children and specialists. The following selection of keys is therefore just a tiny cut-out of interesting examples. As I am German speaking, some of the keys are too.

2.2.1 Books

Books are well-established for centuries and still wildely used. They have to reach a certain quality until they are printed.

Gehölzflora, Fitschen The *Fitschen* (Fitschen, 2002) is the standard key to woody plants in Central Europe. This is not just by chance: It contains a large number (more than 1700) of species, is accurate and well-structured. **Three different dichotomous keys for families** are present: one regarding vegetative characters as leaves, mainly leaves; a winter key viewing for example buds and bark, and an inflorence key. Each family and genus then has a separate key for species; they are sorted alphabetically and can be found easily.

On the first few dozen pages a description is given of most of the characters used in the key. Many graphics support the descriptions as well as the species, as shown in the scan of the *Sorbus* key in figure 2.5.


804 Sorbus

– Bl. etwa um die Hälfte länger als breit, elliptisch bis verkehrteif., 6–10 cm lang, jederseits mit 6–9 Seitennerven bzw. Lappen *(804/1)*, unterseits. graufilzig; Bltn. weiß, etwa 12 mm breit, Gr. 2; Fr. elliptisch, 1,2–1,5 cm lang, ± orangerot, gelbfleischig; V–VI. Bk – Nhk-3. (*S. scandica* (L.) FRIES, *S. suecica* (L.) KNOX & ALMQ.). **Schwedische M., *S. intermédia*** (EHRH.) PERS. 'Brouwers': Pfl. mit geradem durchgehendem Leittrieb; relativ industriefest, als Straßenbaum geeignet.

10 (1). Blch 17–35, je 1,5–4 cm lang 16
– Blch. 7–17, je 2–10 cm lang 11

11. Bltn. etwa 1,5 cm breit, Gr. 5; Bltnst. mit 6–12 Bltn., breit pyramidal, bis 10 cm breit; Fr. 2–3 cm lang, gelbgrün bis bräunl., rotbackig; Blch. (11–)15–17, längl., 3–8 cm lang, gleichmäßig gesägt *(804/2)*; Borke größerer Äste schuppig rau; Winterkn. grünl., wenig behaart, klebrig; V. Bm – Ns-3. **Speierling, *S. doméstica*** L.
Die Fr. werden mancherorts bei der Apfelweinbereitung verwendet; daher in mehreren Sorten mit apfelf. (var. ***pomífera*** HAYNE) bzw. birnf. (var. ***pyrífera*** HAYNE) Fr. angebaut.

– Bltn. meist nicht über 1 cm breit, Gr. 2–4; Bltnst. mit 20 bis vielen Bltn., zieml. flach, bis 15 cm breit; Fr. rundl., höchstens 1,5 cm dick, ± leuchtend rot 12

12. Winterkn. ± dicht weißfilzig behaart, nicht klebrig; Blch. 9–17, längl. lanzettl., 2–6 cm lang, bis auf das untere Drittel scharf gesägt *(804/3)*, obersts. dunkelgrün, untersts. bläul., anfangs behaart; Bltn. etwa 1 cm breit; Fr. bis 1 cm dick, orange- bis scharlachrot; V–VI. Bk/Bm – Bh/Nhk-3. **Nordische E., *S. aucupária*** L.

804/1: Sorbus intermedia | 804/2: S. domestica, Blatt und Blättchen | 804/3: S. aucuparia, Blatt und Blättchen

Sorbus 805

Mehrere Zier- und Fruchtsorten sind in Kultur, z. B.: 'Edulis' (var. *moravica* ZENGERL, var. *dulcis* KIRCHTZ.). **Süße E.:** Fr. bis 1,5 cm groß, süß, schmeckend; Blch. 4–7 cm lang; angebl. in Mähren gefunden, früher gelegentl. wegen der Fr. angebaut; eine ähnl., aus Russland stammende Sorte ist 'Rössica'.
'Dirksenii': Bl. im Austrieb goldgelb, später vergrünend.
'Xanthocárpa': Fr. orangegelb.
'Péndula': Zw. herabhängend (meist hochstämmig veredelt).
'Fastigiata': Wuchs steif aufrecht, ähnl. einer Säuleneiche.

– Winterkn. braun, kahl oder locker behaart, klebrig 13

13. Blch. (11–)15(–17), obersts. blaugrün, untersts. blass, elf. bis längl., 3–7 cm lang, spitz oder kurz zugespitzt, Rand gesägt mit nach oben gebogenen Zähnen, Rhachis und Mittelrippen meist rötl.; Blnst. meist bis zur Fr.zeit behaart bleibend; Blnt. 8–10 mm breit; Fr. bis 1 cm dick; Winterkn. meist locker ± braun behaart; V. Sg/Bk – Bh/Nhk-2. **Labrador-E., *S. decóra*** (SARG.) SCHNEID.

– Blch. obersts. heller oder dunkler frischgrün (höchstens untersts. blass blaugrün), ± lang zugespitzt, Rhachis oft etwas grünl. geflügelt; Blnst. zur Fr.zeit meist kahl; Winterkn. oft glänzend, kahl oder spärl. behaart 14

14. Blch. 3,5–5 cm lang, zu (9–)13, elf. bis lanzettl., scharf gesägt *(805/1)*, obersts. glänzend, Herbstfärbung leuchtend mahagonirot; Blnst. zur Blütezeit meist weißl. bis bräunl. behaart; Krbl. zurückgeschlagen; Fr. 6–10 mm dick, oft orangerot; V. Sg/Bk – B/N-4. **China-E., *S. serótina*** KOEHNE

– Blch. bis über 7 cm lang, zu 11–17; Blnst. meist schon zur Blütezeit kahl 15

15. Blch. 11–17, längl. bis lanzettl., 4–10 cm lang, meist mehr als 4-mal so lang wie breit; Blnt. 5–6 mm breit, Krbl. ± längl.; Fr.

805/1: Sorbus serotina, Blatt und Blättchen | 805/2: S. scalaris, Blatt und Blättchen | 805/3: S. koehneana, Blatt und Blättchen

Figure 2.5: Extract from the Gehölzflora; key for the *Sorbus* family. The bold numbers to the right lead to the next question.

Bestimmungsschlüssel zur Flora der Schweiz This key (Hess et al., 2010) is specialised in Switzerland's flora. **Numerous clear graphics** serve as additional description of the species, as seen in the scanned figure 2.6.

For non-experts it is much harder to use than *Gehölzflora* as there is no entry point with vegetative characters; the root of the dichotomous key mainly considers the inflorescence and uses many technical terms average users are not familiar with, so they are stuck at the very first point of the key.

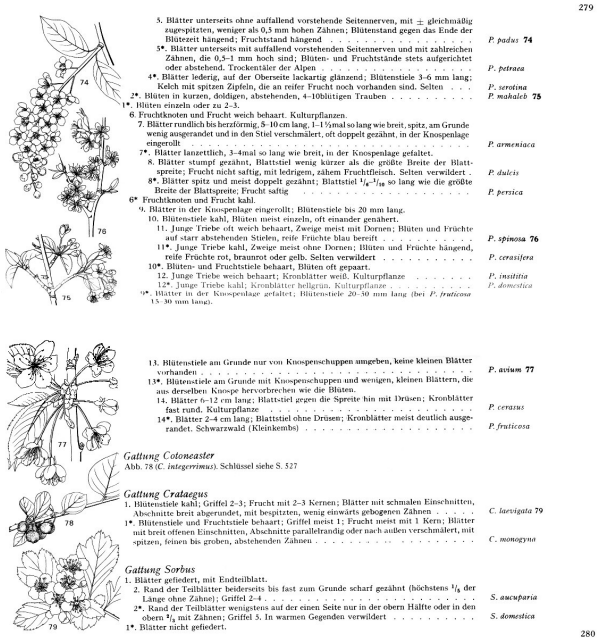


Figure 2.6: Extract from the *Bestimmungsschlüssel* to *Sorbus*

Bäume von A–Z This book is not a key, but only describes various trees. This is done both with text and, especially, a large number of **high-quality and expressive images** showing both macroscopic and microscopic scales, as seen in figure 2.7. They give a good visual impression of the trees' character, i.e. their silhouette, bark, leaves, and other typical characters.

Interestingly, it lacks an index for German names. The trees are sorted by their Latin name, and so is the list at the end of the book, listing additionally names in other languages like German, French, and Spanish. To look up a tree by its German name, one has no other way but to scan this whole list, or to look up the Latin name in a different book.

Figure 2.7: Extract from *Bäume von A–Z* on *Sorbus*

2.2.2 Web Keys

Web keys benefit from links, newer ones are interactive. Countless keys can be found, high-quality keys are more rare though.

People have started early to use the web for identification keys, some count 30 years already. A huge number can be found nowadays. A Google search for "identification key" found 299 000 matches on April 27th 2013, and 315 000 five months later. The number of keys does not grow as fast anymore nowadays; Walter and Winterton (2007) measured an increase from 45 000 to 149 000 in one single year back in March 2006.

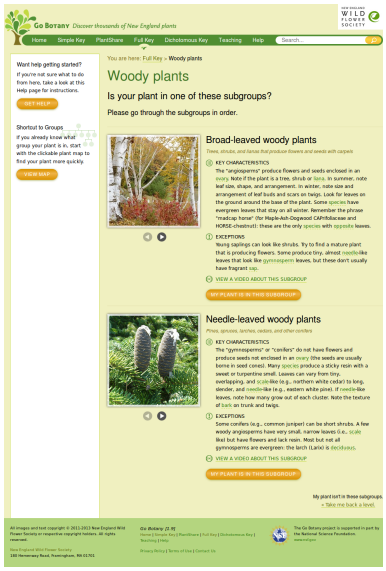


Figure 2.8: Go Botany: Starting page for woody plants

`http://gobotany.newenglandwild.org/` Perhaps the most beautiful key available today. Unfortunately it is hardly visible on search engines. All the more was it interesting for me to find in it nearly all the points that I regarded important for identification. The reader is at this point asked to take a look at the web site.

The entry point of the *Simple Key* is well arranged, on pastel colours, and asks the user to chose woody, aquatic, grass-like, and other plants. For each of them an image gallery, which can be scrolled through, shows representatives, a short **text describes the group, and hints about possible misidentification.** For example, the entry about aquatic plants states: *"Some land plants can be flooded temporarily but cannot live long in water."* They even have a video for each group.

When the group is chosen, subgroups appear (broad-leaved and needle trees for woody plants) in the same manner, leading then to an image gallery of all plants in this group. Simple criteria—only around 8 in the basic view—filter this list. Usually some species remain in the list, but they can be distinguished visually. The species description then lists the characters, shows several images, gives a description of the species and its distribution in England and North America.

To date, the tree list contains 573 species, and has doubled during the last six months.

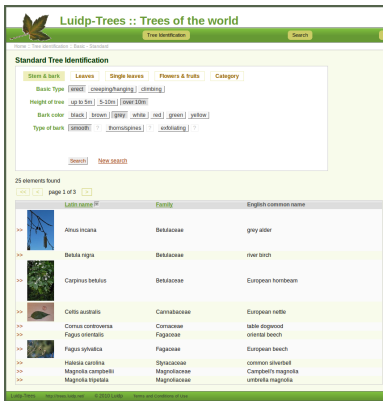


Figure 2.9: Luidp-Trees: Tree identification with a filter

`http://trees.luidp.net/de/index.php` Another web page using filters, the first one of their kind I found. The **filtering system is fast:** Switching groups (Stem & Bark, Leaves, Flowers etc.) works via mouse-over, the view is compact. A click on the search button lists all matching species (unless they are too many).

The key features a fixed number of characteristics. It is easy to get used to them after a few identifications. However, they are sometimes not precise enough, for example when identifying conifers, no further description of their needles can be chosen.

To date, the tree list contains 558 species.

`http://offene-naturfuehrer.de/web/` This web site is unique in that it is the only one made for **public collaboration.** Keys, e.g. in PDF format, can be shared, other keys can be edited directly in a wiki. Those are platform independant and also offered in apps for mobile systems, displayed in form of web pages.

While the idea is great, I got the impression that editing is not as easy as it could be—or should be in order to attract a larger number of editors—as they are hand-written and not generated, making changes more labour-intensive. The web keys are dichotomous and static.

2.2.3 Other Keys

This section shows examples of two different key concepts not further covered here.

LeafSnap A different approach is taken by the authors of Belhumeur et al. (2008). They built a **computer vision based system** to identify plants by an image of the shape of their leaves. This is a very convenient way since the identification process requires no work from the user except for taking an image and looking through the best-matching species (according to the authors, the correct taxon is among the top 10 species returned in 97 % of all cases for 245 species).

This kind of system is obviously not suited in winter and difficult for identifying trees with small leaves (especially conifers with their narrow needles and microscopic details). Also, it does not explicitly describe the characters used for identifying the taxon—it contains machine-readable patterns instead of human-readable facts. To illustrate this point, let's take a key for architectural styles. The key would only tell me that this is a Gothic building, but not that the pointed arch is a typical feature of this epoch. The user can look it up, but does not need to, so it is less "educational" concerning teaching the *whys*, what is special for this taxon or epoch.

The key does not need to do this, however, since its goals are quick and easy identification. The web site mentioned in their paper (`http://herbarium.cs.columbia.edu/data.php`) was never accessible while I was working on the thesis, but they provide an iOS app on `http://www.leafsnap.com`. The user has to take a picture of a leaf on white background, which is then uploaded to their server and rated against existing entries. I could not validate their results since the database contains trees native to US; still, the genus of the top ranked tree used to be correct. The white background sometimes created difficulties when the leaf was larger than an A4 paper, or when images got rejected due to automatic brightness adjustments that could not be controlled manually with the used iPhone.

Bird Songs Yet another approach is taken by audio "keys", commonly found for bird songs. Their principle resembles to image lists, but are also used for learning the sounds by heart and hereby building a **"mental" identification system.**

3 Proposed Solution

3.1 Schema

This section explains the structure and the reasoning behind the database schema used for this identification key.

The "units" that describe a taxon are **characters.** Multiple taxa can share the same characters; the less characters two taxa have in common, the easier it becomes to distinguish between them.

Taxa are only defined by the characters they show, and not by all the characters they do now have; section 3.4.1 gives more details about this choice.

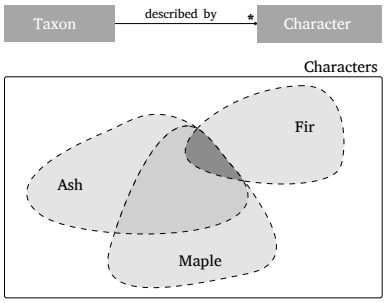


Figure 3.1: Taxon and characters. Ash resembles maple more than fir; they have more characters in common.

It is convenient to group multiple characters in **questions** when they describe different distinctions of the same thing.

The first benefit of doing so is the structure and order character groups bring—their addition cleans up the mess in the—by tendency large—set of characters, and adds some simplicity hereby.

The second benefit is exclusion: If the user observes character A and a taxon shows character B, we cannot conclude *anything* helpful for identification; maybe the taxon actually shows A as well, but the data has not been recorded in the database yet. However, if A and B are distinctions of the same question, we are (usually) on the safe side to assume that the taxon does not show character A.

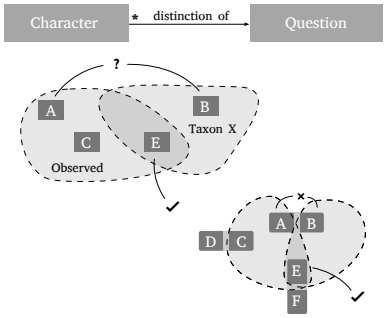


Figure 3.2: Questions and characters. Assigning A and B to the same question allows to conclude a mismatch.

Assigning questions into **components** does not directly have a semantic meaning, the focus now lies on usability. A component can be, in case of a tree, for example the leaf, the bark, or the inflorescence. Grouping questions about the same component helps the user focus on one thing after the other, and skipping all questions about other components manually when e.g. only a leaf is available falls away.

Component grouping also allows to insert contextual information about the current component, for example a leaf or a bud, like a sketch with a legend naming the different parts (leaf stem, blade, veins, edge, etc.). Figure 3.14 shows an example of such contextual information about the leaf.

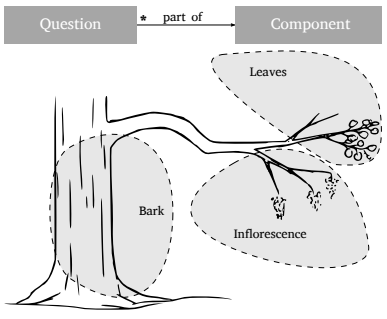


Figure 3.3: Grouping questions into components also allows to display contextual explanations.

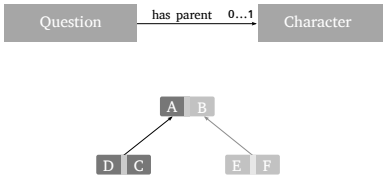


Figure 3.4: A hierarchy for questions allows to uncover relevant questions only when necessary.

While the characters are structured now, we still throw all of them at the user at once, which is painful for them. Introducing **parent characters** for questions shows the relevant ones only. For example, the question whether the needles are bundled or single is relevant only for coniferous trees with needle-like leaves, not for broadleaf trees.

Altogether this leads to a quite simple core schema that is applicable for many topics and not only trees. But it is amazingly powerful!

3.1.1 Generality of Keys

This section extends the schema to support different subjects and discusses differences between topic specific keys.

The basic concept of a key does not change with the subject to identify. Be it plants, fishes, airplanes, all of the three introduced key systems—image list, dichotomous, diagnostic—can be used: **only data differs; information does not.** The dichotomous plant key asks for the leaf type, the airplane key asks for the engine type. This is useful when writing software—once the functionality is there, it can be used for any subject.

One aspect does depend on the subject, though: it is the probability of the item showing a character. Take a key for diseases and a key for light bulbs. An incandescent light bulb will always contain a glowing filament wire. But Lyme borreliosis may cause any of a circular rash, headaches, muscle soreness, or other symptoms, but none of them always occurs. Different rating mechanisms are required for such subjects; an item cannot be excluded when a character cannot be observed. Dichotomous keys are by their nature not suited for this task. Diagnostic keys, on the other hand, are easier to adjust accordingly.

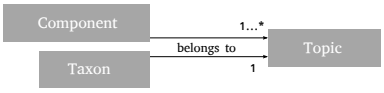


Figure 3.5: Multiple topics can be supported in the same database.

To support multiple **topics** in the schema, let us introduce a *topic* entity. Each taxon is part of a topic, and so are components.

One could argue that assigning topics to components be redundant since they could be deduced by looking which taxa the characters in the components are assigned to. The assignment though represents reality—a light bulb does not have leaves—, and it also simplifies editing a taxon since characters to non-related topics are not shown.

3.1.2 Add-Ons

This section discusses how the g eneric identification key can be extended for special needs of other topics.

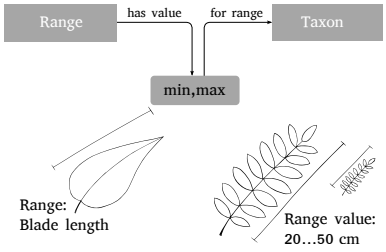


Figure 3.6: Ranges with corresponding range values

With the core schema defined, one can now start adding topic specific extensions or others that are useful. Examples are (partly discussed later): difficulty for questions; taxonomic degree for plants and animals, together with the according higher-level parent taxon; or required equipment for questions. All of them are implemented in the code.

Another useful add-on are ranges for numerical values, since those cannot be represented by characters in a good way; especially for floating-point values used with distances and such. Ranges cannot be represented by a "has" relation directly; the value for the respective range needs to be stored as well.

3.2 Optimising Identification Keys

3.2.1 Choosing Questions in Dichotomous Keys

This section discusses aspects that can be used in order to decide which questions to ask first to keep the identification process as short as possible.

Literature concurs that dichotomous keys should be optimized, but is divided about how this is to be done. Payne and Preece (1980) present in their paper methods to speed up the identification process, for example by minimizing the average path length in dichotomous keys. In this paper's discussion, Dr Sviridov points out that not only speed, but also the probability for identifying the taxon correctly, is of importance—and cannot be optimized simultaneously, as for higher certainty usually more tests are required.

Dichotomous keys can be optimized in terms of maximum depth by choosing characters that separate the remaining taxa into chunks of even size. Yet also the opposite way can make sense: Short paths for common/frequent taxa, long paths for rare ones.

3.2.2 Choosing Questions in Dynamic Diagnostic Tables

This section lists possible methods to "tune" dynamic diagnostic tables, i.e. speed up the process of identification.

Optimising dynamic diagnostic tables—which present several questions at once—requires a different approach since the user can answer them in arbitrary order. The difficulty then is rather the large number of questions: there may be hundreds of them, and endless lists are difficult to overview. So it makes sense to both filter and sort the character list.

Filtering removes questions from the list because they are irrelevant, or not yet relevant.

- **Dependencies on other characters**—For coniferous trees for example, broadleaf characters like the shape of the leaf rim can be hidden. The proposed schema supports this with the parent character of questions; the question is only shown when its parent character is observed.
- **Useless characters**—If answering a question does not exclude any taxa still in the list because they all show the same character, it does not need to be asked. The question can, however, provide additional certainty that the observed taxon is actually the taxon returned by the identification key by over-describing it. Otherwise, having only one matching taxon left—and therefore no additional questions asked—does not necessarily mean that this taxon is the correct one: it could as well be that the correct taxon is not even recorded in the data set, and the questions answered so far happened to match another one.
- **Existence**—A character may not be visible because it simply currently does not exist. Examples are leaves of deciduous trees in Winter, bark on a young tree (as Dr Atkinson pointed out in Payne and Preece, 1980), buds in spring, or the inflorescence during all but a few days of the year. These questions can be skipped. In the proposed schema this is done by hiding components.
- **Equipment**—Answering a question may require special equipment. For example, recognising the shape—or even the existence—of hairs on leaves requires a loupe, and if none is available, then those characters need not be shown.

Sorting has the same goal as optimisation in dichotomous keys; the user should **answer as few questions as possible** for an identification. The common way of sorting questions is sorting them according to a cost function; several of them have been proposed around the 1970s for computer programs, the first ones by e.g. Pankhurst (1970) and Dallwitz (1974), and examined e.g. in Gower and Payne (1975). Only recently have Reynolds et al. (2003) proposed a different measure estimating the amount of work done by, instead of the cost of, answering a question:

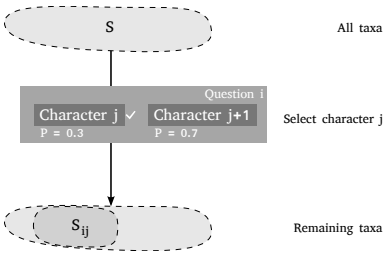


Figure 3.7: Remaining taxa S_{ij}

$$W_i = E(S) - \sum_j P(i,j)\,E(S_{ij})$$

where S denotes the set of Taxa and S_{ij} the remaining set of taxa after answering question i with character j (see figure 3.7), $E(S)$ is the estimated cost of completing identification for the given set, and $P(i,j)$ is the probability of answering question i with character j. For estimating $E(S)$, the authors of Reynolds et al. (2003) use Shannon entropy $H(S)$.

Additional sorting criteria may include:

- **Difficulty**—When generating a key for inexperienced users, questions that can be answered easily can be prioritised. This should also reduce the probability for errors: for example, depending on the location of a fir, it is not easy to see if the twig is haired or if it is just dirt. Answering difficult questions often require more time as well. Difficulties are currently supported but not actively used.
- **Groups**—By grouping characters concerning the same part of a tree, for example a leaf or a bud, contextual information can be inserted, like a sketch with the different characters explained.

3.3 Design Choices

During entering data for the identification key for trees I have gained experience about what works well and what does not. Many, if not all, resulting decisions are generally applicable and not only for trees/plants.

3.3.1 What is a Character?

This section describes how characters should be chosen in order to maintain simplicity—despite the plants showing great variability.

Figure 3.8: *Metasequoia glyptostroboides*. The green shoots fall off in one piece in winter, similar to compound leaves, and the needles are soft.

Deceptive looks Is *Metasequoia* a confier or not? The leaves are soft, and they resemble mimosa leaves, not conifer needles. The question *"is it a soft- or a hardwood tree?"* (*softwood* trees are conifers, *hardwood* trees are the others) is not easy to answer if one does not know the tree already. The question is therefore likely to be answered incorrectly for *Metasequoia*.

One could now simply tag the tree both as soft- and as hardwood, and the user could identify it on both ways. This, however, is not correct. The tree is not a hardwood. Or, one could tag the character for this taxon with "could mistakenly be answered with *hardwood*". Which gets complicated.

Another example is the *Podocarpus* genus. The trees have broad leaves, but belong to the conifers.

The better way is to ask: *"Are the leaves needle-like or broadleaf-like?"* With good conscience can we say yes to both.

Describing what something looks like is easier than naming it. Otherwise, why would we need identification keys?

Figure 3.9: *Prunus spinosa*. Young individual on the left, mature one on the right. Their leaf shape is nearly the opposite.

Handling variability *Prunus spinosa* has reverse egg-shaped leaves. If the tree is young, however, they are egg-shaped (not reverse!). *Fraxinus excelsior* has a smooth, greyish stem. After 20–30 years it develops a thick bark that breaks in longitudinal lines.

At first glance it looks like conditional characters were a good idea: *If* the tree is young, *then* the leaves have this shape. However, what is young? Such **conditions are hard to determine,** and show as much variance as the characters themselves.

My solution is again the simple one: The tree can show both characters, so both of them are set for it. If only one character is observed, the correct tree is still found.

3.3.2 Exclusiveness of Characters

This section shows why allowing at most one answer to some questions does not make sense.

I thought several times about adding an *exclusive* property to questions, implented it, and finally decided to drop it.

Figure 3.10: Alternate and opposite leaves

Exclusiveness—allowing only one answer for a question—intuitively makes sense. Leaf branching serves as example: One distinguishes between opposite (pairs of leaves are attached at the same height on the twig) and alternate (one left, one right, etc.). Maple leaves are opposite, beech leaves are alternate. Tagging this question as exclusive would disallow the user to select both answers, which would not make sense anyway.

Unfortunately, that is wrong. Some willows have opposite *and* alternate leaves.

Some characters, however, *are* exclusive. All Beech species have the same kind of fruit, as do *Castanea* species. But also this exclusiveness is lost when adding not only species to the key, but also genera, families, and further higher-level degrees: Both belong to the family *Fagaceae*.

Consequence is that **exclusiveness does not work.**

It is important to note that leaving away exclusiveness does not weaken the key; wrong answers are still counted as mismatch.

3.3.3 Numerical Values

When dealing with numerical values, ranges are required since nature, as well as human measurement, usually shows variability.

More than once has it happened to me that I tried to identify a shrub with leaves around 1 cm long, and ended up identifying it as a tree whose leaves are usually over 30 cm long. The reason was that the dichotomous key only asked for size independent characters like the leaf shape or the nervature type. The possibility of specifying the length of the leaf would have prevented this misidentification.

Plants typically show a **great deal of variance especially for dimensions.** Petitoles of *Acer* easily range between 5 and 20 cm on a single tree; usually they are longer the lower on the branch they grow, to maximise light yield. Also growth of the tree itself varies: tree rings of yews can be observed from 0.1 mm in alpin regions up to over 10 mm in gardens.

Those examples make clear that numerical values must be represented as ranges. This is even the case for longer values, e.g., the number of leaflets on a *Fraxinus excelsior* leaf ranges between 11 and 15 in general.

To keep the variability within reasonable limits, extreme values should be ignored. For leaves one can almost always find an even smaller leaf; bonsais for example are miniaturised in every regard. Leaving away extreme values does not decrease the chance of identification in general; the user would very likely not pick the ash leaf with the most leaflets, but chose an average number. For bonsais or other extremes, they have to be (made) aware of the resulting changes.

One of the few cases where numerical values can be replaced by characters are conifers: If they have bundled needles, typical values are only 2, 3, 5, and "many" (more than one wants to count).

Testing an observation for a hit is done by testing if it covers or intersects with the range stored to the tested taxon.

3.3.4 Geographical Distribution

This section explains why geographical distribution maps are not supported.

Many keys or books feature geographical distribution maps. I thought a long time about the best way of implementing them. What should be the units? Country borders change. Grids may be tgoo coarse or too fine-grained, depending on the taxon and the landscape. Sample points are accurate but only at a single point. And finally, the locations themselves change with plants transported to and planted at the other end of the world.

I ended up with the conclusion that the best way of implementing geographical distribution maps was *not at all.* It is the wrong model: plants are not hindered from **spreading across boundaries defined by GPS coordinates:** *Metasequoia* originally lives in China, but grows just as well in Switzerland. The correct model is the habitat: plants require a certain temperature range, precipitation level, pH range, and others.

3.4 Decisions for Key Generation

The identification key is **character based;** a taxon can be identified by examining it accurately.

3.4.1 Does This Taxon Match?

This section explains the rules used for deciding when a taxon matches the observed characters.

Matching is done by comparing the observed values to the stored data—not the other way round.

My proposed method for **rating taxa** counts positive, negative, and unknown matches. A *positive match* is clear: if we observe a character the taxon shows as well, the match is positive. If the taxon does not show this character, we have either a negative match or we have not enough information.

A *negative match* occurs only when the observed character is not observed for the tested taxon *and* the tested taxon shows a different character for the same question. If the tested taxon does not have any character defined for the question in question, we cannot tell if it matches the observation; either the question does not apply (e.g. leaf shape for needle trees), or the data has not been entered yet.

One could argue that we may know the parent character of a question, and hereby use the question tree to invalidate more characters. In case of testing against a fir, we know that if a leaf shape is specified, we have a negative match; the leaf shape is a child of the *broadleaf-like leaves* character. My experiments have shown that this excludes taxa far too quickly in case of a misidentification (or of wrong data) since one single observation can affect several—many—characters at once.

The ruleset thus stays very simple, and taxa remain described only by the characters they show, which, in my opinion, is a clean and meaningful way. Requiring to list all characters that do not match as well would lead to a trememdeous amount of work when introducing new characters, and in effect duplicate data, which should generally be avoided.

Complexity of this approach is $\mathcal{O}(c\,T)$ for T taxa and an average of c characters per taxon.

The most simple way for **excluding a taxon** is to exclude it as soon as an observed character does not match the characters stored for it anymore. As long as both data and observations are correct, this is the fastest way for identification.

Allowing for a certain number or percentage of errors makes sense if chances for misidentification are high, which is especially the case for difficult trees like the *Cupressaceae* family.

3.5 Implementation Details

3.5.1 Identification Key

This section discusses languages and patterns used for implementing the identification key.

Together with the data model, the identification key builds the core part of my thesis. What are the best suitable technologies for implementing them?

For the implementation of the identification key I first had to answer the classical question which programming language I wanted to use. The decision fell on a web-based JavaScript application. The reasons are the following. First, platform independence is simply not an issue anymore—browsers supporting **JavaScript and HTML5** are available on all major operating systems, including mobile ones for phones and tablets. Second, no installation is required. Third, no server communication is necessary anymore after the key has been loaded.

The third point is realised by the usage of a fat client—i.e. the JavaScript client contains all logic for identification—, and by loading all required data on initialisation.

Two important core libraries are used by the key: the data model library and the identification library. Both of them are re-used in the editor, as explained in the next section.

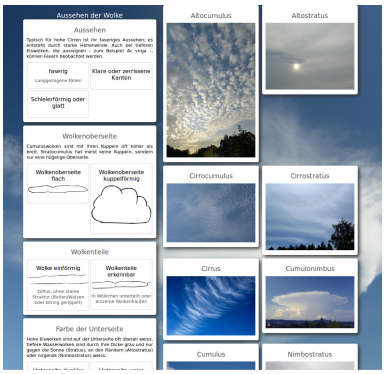


Figure 3.11: Identification key UI for clouds. The remaining taxa scroll together with the questions, but stop early enough so they never float out of the viewport.

The identification key is constructed in a configurable queue of functions that process both the taxon and question set. Taxa are processed first; their rating is required for sorting questions afterwards.

Taxon processing is accomplished in three steps. In the first step, prefilters exclude taxa that are not interesting to the user. For example, only species are shown for trees, since genuses and other taxa of higher degree are merely used for constructing a taxonomic tree; they do not have any characters assigned. Or, the user might be interessted in species of the *Cupressaceae* family only; all others are then filtered. The second step is assigning the ratings to the set of remaining taxa, i.e. counting matches, mismatches, and unknowns for the current observation. This information is then used in the third step for filtering out taxa that do not match the observations and for sorting the others.

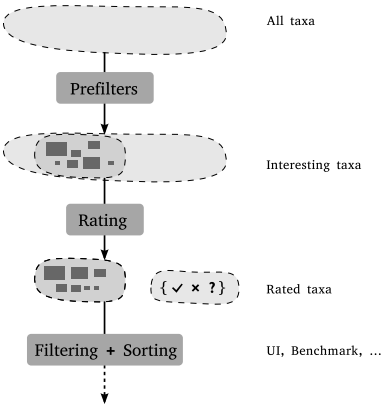


Figure 3.12: Taxon processing queue used for the taxon thumbnails on the right-hand side in figure 3.11.

Question processing works in a similar way. The prefilters can again remove questions that are not interesting; currently only one is available, hiding questions if required equipment, like a louope, is not available to the user. The second step, rating the questions, calculates basic statistics like how many times a character occurs in the set of remaining taxa; they are then used by one of the cost functions, which themselves provide functions for sorting taxa according to the costs they calculated, or by other ways. The filter, finally, e.g. decides if questions should be hidden when they cannot help in narrowing down the set of remaining taxa, or not to allow over-describing taxa for increased certainty.

With those two queues completed, the user interface is constructed using a visitor pattern for both taxa and questions. Question visitors can optionally use components for better overview, as seen in the screenshot 3.14, or disable them for better performance—there is only one top question and not one per component—, and use depth-first search to keep questions with parent–child relationships together.

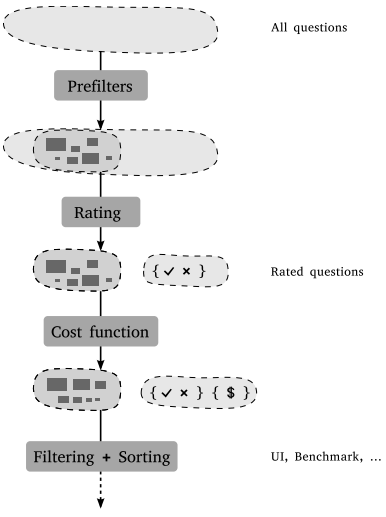


Figure 3.13: Question processing queue (questions on the left in figure 3.11

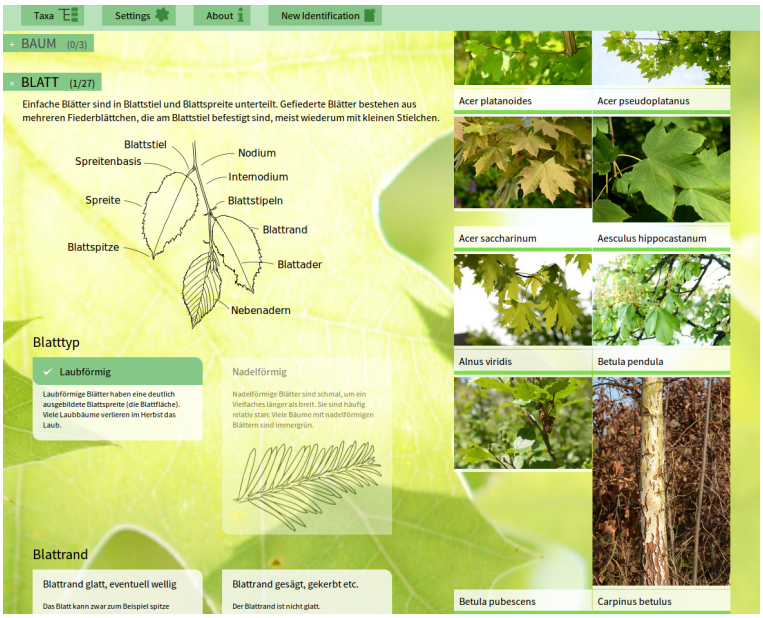


Figure 3.14: Main UI; the first component *Baum* is hidden. The numbers give the questions answered and available.

For the design of the user interface, my goals were to

- show **relevant information** only. Settings, which usually are not changed often, are accessible in the menus. Taxon descriptions are provided when clicking on their image; the thumbnails can be scanned quickly by our brain for validation, text however cannot.
- use **space** efficiently. The key contains a possibly vast amount of information, and displaying it clearly requires a combination of space-saving design and hiding of irreleveant information, while ideally keeping the visual impression lightweight.
- maximise **identification speed.** Hand-drawn graphics explain the characters described in questions, allowing new users to soon answer questions based on the graphics without reading the explanation—which is much faster. The set of remaining taxa never scrolls out of sight, so the user does not need to scroll back in order to see how many taxa are remaining. Uninteresting components, like leaves in winter, can be hidden so the relevant questions are found quicklier.

The user interface for trees supports an additional feature, a taxonomic tree. If items are selected, only they—or their children—will be displayed. Experienced users need this as they can often tell e.g. the familia of a taxon due to similarities with other species, and can hereby quickly exclude all other taxa. An example can be seen in figure 3.15.

3.5.2 Key Editor

This section descusses the two backends for the editor that are currently available.

The editor is built of two parts, the front-end user interface where the actual data is entered, and the back-end storing the data. For convenience, the user interface is—like the identification key—written in JavaScript and HTML5.

As mentionend before, the identification library is re-used for the editor: it provides life updates on the taxa most similar to the one that is currently edited. When adding characters to the taxon, the library functions again hide ones that do not come into question, e.g. due to parent–child relationships.

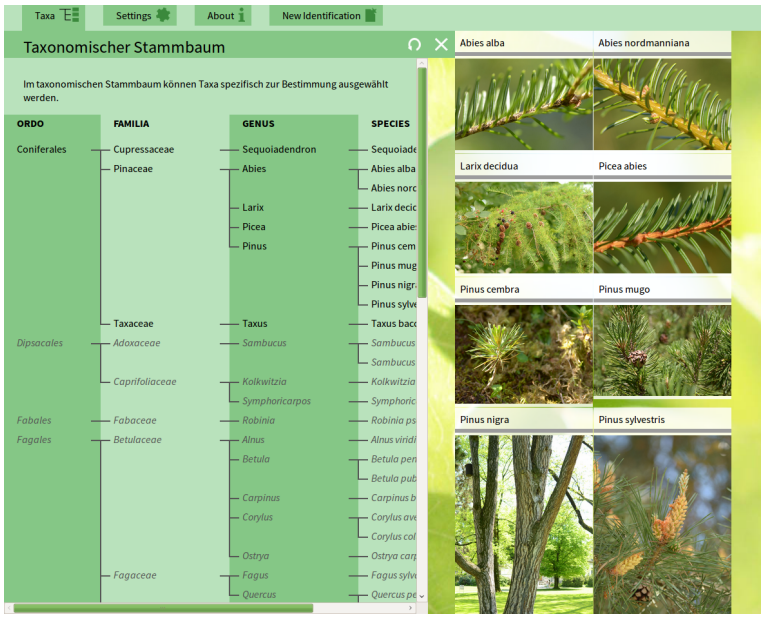


Figure 3.15: UI with the taxonomic tree; *Coniferales* are selected, other taxa are hidden.

The first of the two back-ends available stores the schema directly in a MySQL database; it is written in PHP, and the communication with the front-end happens in Ajax/JSON. No additional information (metadata) is recorded.

The second back-end is more interesting. It is a Mediawiki extension using Wikibase. The reader is probably familiar with Mediawiki as it is used on Wikipedia: it is a multi-user content editing system with history support, similar to a version control system. Wikibase now is an extension to Mediawiki and transforms it into a **key/value database.** More precisely: it consists of properties (keys) and items. Items have an ID, a name, and property/value pairs; values may be, for example, strings, images on Wikimedia Commons, or again items. Properties have an ID and a name.

So far, this adds support for multiple users and a history, compared to the first back-end. Yet the most important point is: all data is translatable. The point mentioned in the goals—structure should be language independant—is hereby completed.

Ranges and numerical values are not implemented yet in Wikibase, therefore the Mediawiki extension currently does not support ranges.

Wikibase is still in heavy development, and had I started my thesis only half a year earlier, I would not have had the chance to make use of this great tool!

3.5.3 Mobile Identification Key

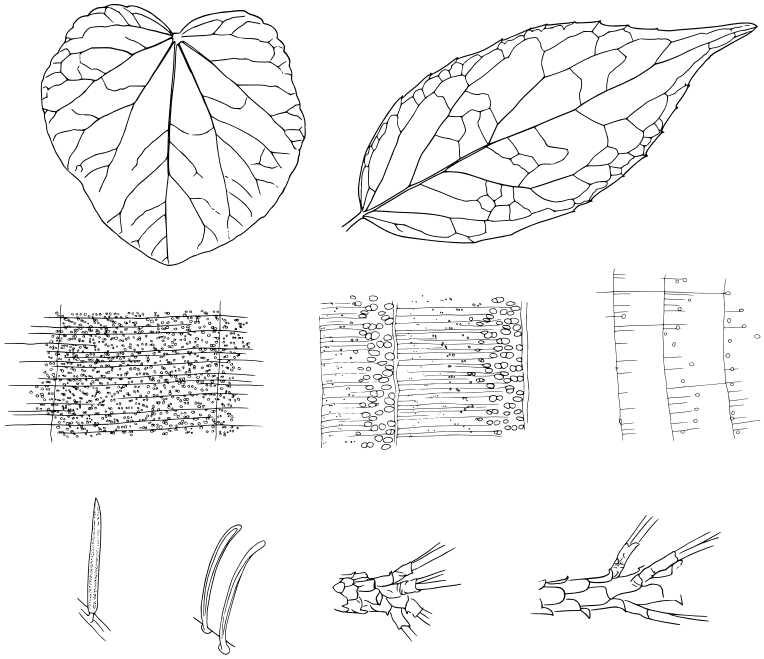
Since mobile devices strongly focus on web and web applications, very little changes are required for a mobile version.

Having an offline version is useful when working in the field. WLAN is usually not available there, and even mobile coverage may be absent in sparsely populated areas.

As mentioned in 3.5.1, the identification key logic runs on the client side and loads a database dump on initialisation with Ajax calls. An **offline version** is therefore very simple to create—the JSON dump, i.e. the responses to said Ajax calls, only needs to be saved in a file and can then be loaded with the same Ajax calls as before.

The app itself is then generated with the Cordova platform[1]. Cordova can build apps for all major mobile platforms from HTML5 web applications, and provides JavaScript libraries to access e.g. the camera of the mobile device.

Current efforts in browser development may soon make this step superfluous. Web applications can be cached by the browser in its application cache *(AppCache)*, and can then be used offline like a normal app on mobile devices. The current user interface already works without changes on tablets, thus using this mechanism should be relatively easy.



[1] Cordova is available on http://cordova.apache.org.

4 Conclusion

4.1 Evaluation

4.1.1 Question Order Benchmark

This section shows experimental results measured for different question rating algorithms, which are responsible for fast identification.

To compare the different ranking algorithms for questions, I wrote a benchmarking script that counts the steps required for identifying a taxon by answering the top rated questions. The algorithms compared are:

- Alphabetic: No rating at all; questions are simply sorted alphabetically by their name.
- Most used: This algorithm takes the set of remaining taxa, i.e. the ones matching the current observations, and counts for each character how many times it is used by any of those taxa. The question rating is then the sum of its characters' counters, and higher is better.
- Entropy: The algorithm proposed by Reynolds et al. (2003) using Shannon entropy to estimate remaining cost

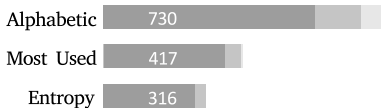


Figure 4.1: Question rating benchmark for 57 taxa and 193 characters, and around 17 characters per taxon.
Light grey: 61 taxa, 205 characters, 19.6 characters per taxon.

Except for one taxon, the entropy based algorithm always performed equally good or better than the most-used algorithm. The results of the benchmark are shown in figure 4.1.

All algorithms have linear complexity $\mathcal{O}(C)$, C being the number of characters, after the question ratings have been computed as discussed in 3.5.1. The complexity of latter is, similar to section 3.4.1, $\mathcal{O}(c\,T + C)$, with the number of taxa T, each having an average of c characters assigned. The overall complexity, including sorting, is $\mathcal{O}(Q \log_2 Q + c\,T + C)$, with Q being the total number of questions. (All terms are independant of each other and therefore cannot be canceled.)

4.1.2 Are The Goals Met?

This section looks at the problem statement in 1.2 and compares the original goals to the resulted system.

I have written a fully working, generic identification key containing data with photographs and hand-drawn graphics for trees and for cloud genera, which is more than I hoped. Additionally, I have a prototype extension for Mediawiki/Wikibase and a prototype Android app.

Editability Works with two back-ends available; Mediawiki together with Wikibase and the LifeWeb extension provides multi-user support and history, and the plain PHP/MySQL backend aims for simplicity and allows compact database backups.

Performance Questions can be sorted such that $\mathcal{O}(\log_2 n)$ is achieved. The user interface also requires a single click for selecting characters and supports graphics for characters, questions, and components (as in the tree key) that give a tremendeous speed-up compared to text only.

Internationality Both user interface and data can be translated; the user interface uses JavaScript based methods, and data is translated by using Mediawiki's and Wikibase's built-in multi-language support.

Growth All code is released under the Open Source license GPL, and the identification key for trees contains data to around 60 trees, covering already a good part of the trees native to Switzerland. Only future can show if the project manages to grow by itself. The Wikidata.org integration will first require some work, as discussed in the next section. As for code quality—most code has been refactored or re-written at least once, which simplified and improved the code.

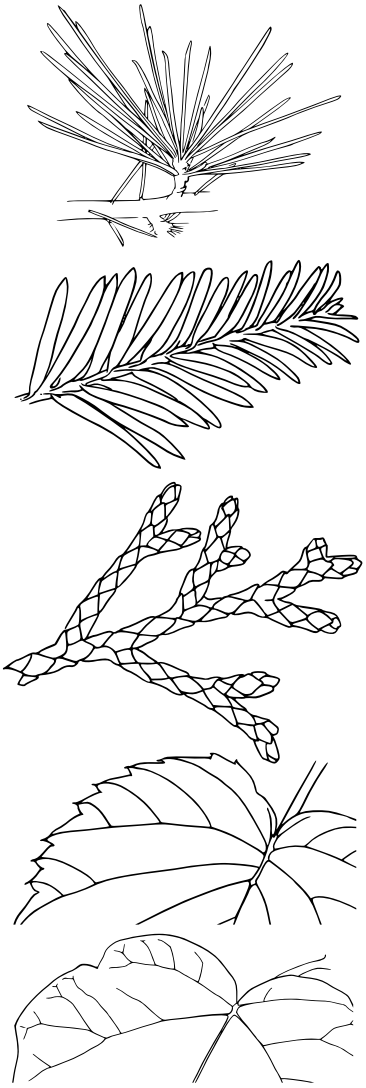


Figure 4.2: Some SVG graphics from the identification key.

4.2 Future Work

This section discusses tasks I regard as important for future work on this identification key.

Ranges or numerical values are not implemented in the Mediawiki plugin since Wikibase does not support them yet. Ranges are currently also disabled in the user interface; a fast and comfortable way for entering numerical values spanning various magnitudes (leaf length may range from millimeters to meters), and a meaningful cost function for rating the importance of ranges when sorting questions, needs to be developed.

Collections containing trees at special locations, for example in botanical gardens, need to be re-visited. I have removed them since I was not confident of their usability, especially regarding structuring them. Using data from Wikidata, like geographical coordinates or associated country, would extend the possibilities.

wikidata.org is an ideal place for data and should be a long-term goal for this identification key. The QueryEngine extension for Wikibase may be used in future for querying data; the current way does not support real-time updates with the amount of data present on Wikidata (around 13 million rows as of August 2013).

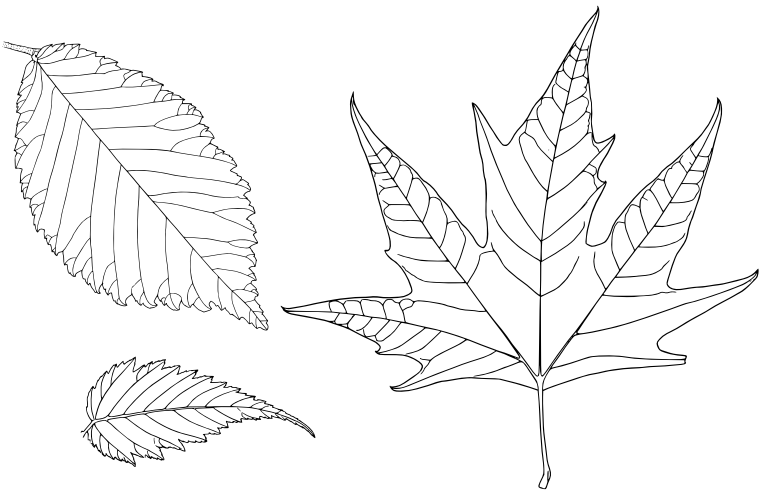
Component based thumbnails for the remaining taxa for visual exclusion. As discussed in section 2.1.1, the human brain is great at pattern matching. Currently, the identification key user interface simply takes the first image as thumbnail. In the example key for trees, visual matching could be supported by taking images of, for example, single leaves of each tree, or their fruits, and only showing those when desired, so that a specific character can be examined on all remaining taxa.

Mobile version neds more treatment; additionally to the points discussed in section 3.5.3—adaption to new technologies like browser based apps—a single page for different screen resolutions is desireable. Currently there is a separate version for mobile phones, while tablets can use the standard web page.

Editor UI should be re-designed. The current user interface is a development version that focuses on functionality only, and usability has been neglected a bit. I have created a user interface design in Inkscape, but it is not implemented yet.

4.3 Acknowledgements

First I would like to thank Prof. Ottmar Holdenrieder and Andreas Rudow: In their lectures I learned new ways to identify plants, and they shared their experience and joy on the topic. Spepcifically, it was a lecture held by Ottmar Holdenrieder in which we used the Fitschen (2002) for identifying uncommon trees for our herbarium where the iea of an electronic identification key came into my mind. Andreas Rudow also helped me with his broader view where I sometimes focused too much on details, and with his own experience in collecting data for trees.



A big thanks goes to Carmen Maria Rovina. We together visited Ottmar's lecture. Carmen accompanied me during the thesis by helping with my time plan, by insisting on me defining the next goals and prioritising them correctly, and with her interest in the topic. She also tested the identification key and proof-read my thesis. As she studied Environmental Systems Science, her knowledge came in handy for topic related questions too. Carmen also managed to locate the LeafSnap page.

Several people tested my identification key and provided valuable feedback: Carmen Ferri, Iris Huber, Katharina Schwitter, Claude Barthels, Anja Taddei, Natalie Kaiser, Gabriela Fisch, Dr Thomas Niklaus Sieber; and Loredana Vamanu lent me a tablet for testing. Thanks a lot!

Then I want to thank my supervisor Prof. Donald Kossmann for supervising this "non-classical" computer science thesis and for his interest in trees, and my Master's Mentor Prof. Markus Püschel for his unique way of presenting complex lecture topics, and for pointing me to the books of Edward Tufte, especially Tufte (1997).

Thanks to my parents for my interesting childhood—for showing us plants, for driving to forests where we could climb on trees, and for hiking—and for their good cooking.

Finally, thanks to the Wikidata developers for their project that suits so well, and to the developers of PhpStorm for writing a great JavaScript IDE.

Bibliography

Mercedes Alsina and Anicet R. Blanch. A set of keys for biochemical identification of environmental vibrio species. *Journal of Applied Microbiology*, 76(1):79–85, 1994. ISSN 1365-2672. URL http://dx.doi.org/10.1111/j.1365-2672.1994.tb04419.x.

Peter Belhumeur, Daozheng Chen, Steven Feiner, David Jacobs, W. Kress, Haibin Ling, Ida Lopez, Ravi Ramamoorthi, Sameer Sheorey, Sean White, and Ling Zhang. Searching the world's herbaria: A system for visual identification of plant species. pages 116–129. 2008. URL http://dx.doi.org/10.1007/978-3-540-88693-8_9.

M. J. Colloff and F. Th. M. Spieksma. Pictorial keys for the identification of domestic mites. *Clinical & Experimental Allergy*, 22(9):823–830, 1992. ISSN 1365-2222. URL http://dx.doi.org/10.1111/j.1365-2222.1992.tb02826.x.

Allen J. Coombes. *The Book Of Leaves*. New Holland Publishers, 2011.

M. J. Dallwitz. A flexible computer program for generating identification keys. *Systematic Biology*, 23(1):50–57, 1974. doi: 10.1093/sysbio/23.1.50. URL http://sysbio.oxfordjournals.org/content/23/1/50.abstract.

Jost Fitschen. *Gehölzflora*. Quelle & Meyer, 2002.

J. C. Gower and R. W. Payne. A comparison of different criteria for selecting binary tests in diagnostic keys. *Biometrika*, 62(3):665–672, 1975. doi: 10.1093/biomet/62.3.665. URL http://biomet.oxfordjournals.org/content/62/3/665.abstract.

Hans Ernst Hess, Elias Landolt, Rosmarie Hirzel, and Matthias Baltisberger. *Bestimmungsschlüssel zur Flora der Schweiz und angrenzende Gebiete*. Birkhäuser, sixth edition, 2010.

J. LaSalle, Q. Wheeler, P. Jackway, S. Winterton, D. Hobern, and D. Lovell. Accelerating taxonomic discovery through automated character extraction. *Zootaxa*, 2217:43–55, 2009. URL http://www.mapress.com/zootaxa/2009/f/zt02217p055.pdf.

R. J. Pankhurst. A computer program for generating diagnostic keys. *The Computer Journal*, 13(2):145–151, 1970. doi: 10.1093/comjnl/13.2.145. URL http://comjnl.oxfordjournals.org/content/13/2/145.abstract.

R. W. Payne and D. A. Preece. Identification keys and diagnostic tables: A review. *Journal of the Royal Statistical Society. Series A (General)*, 143(3):pp. 253–292, 1980. ISSN 00359238. URL http://www.jstor.org/stable/2982129.

Alan P. Reynolds, Jo L. Dicks, Ian N. Roberts, Jan-Jap Wesselink, Beatriz Iglesia, Vincent Robert, Teun Boekhout, and Victor J. Rayward-Smith. Algorithms for identification key generation and optimization with application to yeast identification. In *Applications of Evolutionary Computing*, volume 2611 of *Lecture Notes in Computer Science*, pages 107–118. Springer Berlin Heidelberg, 2003. ISBN 978-3-540-00976-4. URL http://dx.doi.org/10.1007/3-540-36605-9_11.

Edward R. Tufte. *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press USA, 1997.

David Evans Walter and Shaun Winterton. Keys and the crisis in taxonomy: Extinction or reinvention?*. *Annual Review of Entomology*, 52(1):193–208, 2007. doi: 10.1146/annurev.ento.51.110104.151054. URL http://www.annualreviews.org/doi/abs/10.1146/annurev.ento.51.110104.151054. PMID: 16913830.