Retrieving taxa names from large biodiversity data collections using a flexible matching workflow

Edward Vanden Berghe[1], Gianpaolo Coro[2], Nicolas Bailly[3,5], Fabio Fiorellato[4], Caselyn Aldemita[5], Anton Ellenbroek[4], Pasquale Pagano[2]

[1] *Vrije Universiteit Brussel (VUB), Brussels, Belgium,*
[2] *Istituto di Scienza e Tecnologie dell'Informazione A. Faedo, CNR, Pisa, Italy,*
[3] *WorldFish, Penang, Malaysia,*
[4] *Fisheries and Aquaculture Department, Statistics and Information (FIPS), FAO, Rome, Italy,*
[5] *FishBase Information and Research Group, Inc. (FIN), Los Baños, Laguna, Philippines.*

Abstract

In the domain of biological classification there are several taxon name matching services that can search for a species scientific name in a large collection of taxonomic names. Many of these services are available online, and many others run on computers of individual scientists. While these systems may work very well, most suffer from the fact that the list of names used as a reference, and the criteria to decide on a match, are hard-coded in the engine that performs the name matching. In this paper we present BiOnym, a taxon name matching system that separates reference names lists, search criteria and matching engine. The user is offered a choice of several taxonomic reference lists, including the option to upload his/her own list onto the system. Furthermore, BiOnym is a flexible workflow, which embeds and combines techniques using lexical matching algorithms as well as expert knowledge. It is also an open platform allowing developers to contribute with new techniques. In this paper we demonstrate the benefits brought by this approach in terms of the efficiency and effectiveness of the information retrieval process with respect to other solutions.

Keywords: Taxon names matching, Taxonomic Authority File, Taxon Name Parsing, Name Matcher Chain, Taxonomy, Taxonomic nomenclature

Email address: evberghe@gmail.com,n.bailly@cgiar.org,{gianpaolo.coro, pasquale.pagano}@isti.cnr.it,{fabio.fiorellato, anton.ellenbroek}@fao.org,c.aldemita@fin.ph (Edward Vanden Berghe[1], Gianpaolo Coro[2], Nicolas Bailly[3,5], Fabio Fiorellato[4], Caselyn Aldemita[5], Anton Ellenbroek[4], Pasquale Pagano[2])

1. Introduction

"What's in a name?" (Shakespeare, 1599: Romeo & Juliet, Act 2, Scene 2)

Querying that question in Google Scholar[1] just in the "title of the article" field will yield more than 3,400 records (as of November 2014) in a wide range of domains of human activities. It has also been used many times as the title of taxonomists' oral presentations (or as slide title) to convey the important message that the proper management of scientific names of fossil and extant living organisms is essential to the understanding and the management of biodiversity. Coining names for artefacts of the physical world and for human conceptual constructions is essential to our communication. Scientific domains are themselves named xxx-logy, the etymology of the ancient Greek suffix root being "logos" (λογος) meaning a speech/discourse/debate; those would not be possible without names.

In biological taxonomy, the meaning of that question becomes: what is to be known through the scientific name about the organism it designates? The fact is that all data, information and knowledge about species are "hooked" to a scientific name. Therefore, (i) all what we know about a species can be retrieved from the literature by looking for the species name, which can be seen as indexing metadata (Patterson, 2014); (ii) different information systems can exchange data through species names, which can be seen as identifiers.

There should be an unequivocal link between a name and an artefact or a concept. This was clearly the goal when scientific names and their codes of nomenclature were developed. With vernacular names, which usually originate unplanned from common use, this is clearly not the case. However, even with scientific names, this unequivocal relationship is not absolute. Patterson et al. (2010) summarized the main issues that name matching encounters, among them: plain simple misspellings (formally known as "lapsus calami" in the literature on nomenclature), new combinations, several name-as-string variants for one name. These issues make it difficult to use them as identifiers. But they are quite efficient as indexing metadata to retrieve information on species.

To improve the role of scientific names as key to bind information from different sources, it is necessary to standardise their spelling and use. This is achieved most often through a process of matching them with a *Taxonomic Authority File* (TAF), i.e. a list of reference terms, including indication of synonyms and variants of scientific names, their authorship and possibly their data providers. The closer the match the better the chance that both systems speak about the same taxonomic concept. The human brain is quite good at matching names while detecting errors. But electronic systems match the strings of characters that constitute names outside any context, which makes them prone to compute false negatives. One understandable source of mistakes is that colleagues of whom their mother tongue is written in a non-Roman

[1] scholar.google.com

script are more likely to make spelling mistakes. For example, based on our experience and partially supported by the statistics in Froese (1997), the number of misspellings is high in Indian, Arabic, Chinese, and Russian journals.

Matching a string of characters is not enough to understand whether the intended species concept is the same: similar names might cover different species concepts (homonyms), different names might cover identical species concepts (synonyms). Resolving these issues is a different process from the taxon name matching that we will focus on here. Taxon name matching is a necessary first step, before the *content* (or in other words, the taxonomic concept covered by the name) is considered. The second step, concept matching, generally involves expert knowledge, and is considered the role of the Taxonomic Authority File: it is through the TAF that taxonomists have made their expertise available (Lambe, 2014), and allow us to judge whether names should be considered valid or invalid, and to disambiguate homonyms. For example, this approach is evident in the knowledge building process followed by the Catalogue of Life (Bisby et al., 2004), FishBase (Froese and Pauly, 2000) and WoRMS (Costello et al., 2013).

Several taxon name matching systems are available online, and many more are no doubt living on computers of individual scientists; a brief overview of those best known to the authors is included in Section 2, based on our knowledge of the tools used by several scientific communities around taxa matching. While these systems may work very well, many suffer from the fact that the list of names used as a reference (the TAF), and the criteria to decide on a match, are hard-coded in the engine that performs the name matching. The objective of this paper is to describe the BiOnym taxonomic name matching system that separates these elements.

In constructing such a system, it is not always possible to find the one size that would satisfy all the needs; to our experience, in the area of taxon name matching it seems that this "one size" is non-existent. Our ambition was to create a flexible, highly customisable framework to facilitate taxon name matching. This flexibility is deemed important for several reasons. First of all, it is important for determining if the scope of the reference list used is as close as possible to that of the list of names to be tested. For example, if a list of names of fish is compared with a very wide reference list such as the Interim Register of Marine and Non-marine Genera (IRMNG, Rees (2008a)) or the Catalogue of Life (Bisby, 2000), chances are that a lot of near-matches will actually be false positives (or even full matches comparing zoological names against a botanical TAF, and vice-versa). Consider the case of the genus "*Tisbe* Lilljeborg, 1853", a marine harpacticoid copepod. The genus is named after Thisbe, of "Thisbe and Pyramus" fame, but actually misspells the name of the mythological character. The correctly spelled "*Thisbe* Hübner, 1814" is a genus of butterflies. If the name "Thisbe" is used for a copepod, or in any marine context, it is very likely to be a misspelling for *Tisbe*. If it is compared with a TAF of the wrong scope, it might end up as the butterfly. On the other hand, if it is compared with a TAF including exclusively marine names, or with a TAF specific for crustaceans, "Thisbe" would likely be identified as a misspelling of "*Tisbe*". Another example

is reported in Table 1.

Another reason why we need a flexible approach is that the objectives of the end-users are not always the same, and dependent on the "use case". One possible use case for taxon name matching is to suggest, to some end user, a list of alternative valid names, for a list of names (s)he wanted to test. In this case it is important that the "correct" match is in the list of potential matches returned; the fact that other, false matches are also returned is of secondary importance: the "recall" should be as high as possible. Compare this with another potential use case, where taxonomic name matching is used to automate the association of names from a new dataset with names in a reference list. In this case it would be important to have a single suggestion for the matching name - in other words, that "precision" would be as high as possible. For this second use case we can break up criteria even further, according to the weight a wrong match would carry. If, for example, the taxon name matching was performed in the framework of merging different biogeographic data sets, the number of false positives should be weighted against the number of distribution records that cannot be used because no match was found. If, on the other hand, the taxon name matching was performed in the framework of the completion of a taxonomic reference list, false positives carry a much larger penalty, and should be avoided as much as possible.

Thus, in the first use case, it will be important to have a "recall" that is as high as possible; in the second use case, the "precision" will be the most important criterion. Recall and precision, and other measures of the quality of the matching process, will be further discussed in Section 5.1.

This paper is organized as follows: Section 2 reports an overview about taxon name matching. Section 3 explains our approach step-by-step, from the general rationale to the technical details. Section 4 explains the format of the reference datasets used by our process to search for the correct transcription of a species scientific name and the test dataset we prepared to evaluate the performance of our system. Section 5 reports the evaluation of the performance of each component of our method, both in terms of efficiency and effectiveness. Finally, Section 6 draws the conclusions.

2. Overview

Lexical matching is a standard computer application that crops up in several circumstances, for example in the spell checker of a word processor. Many general-purpose algorithms have been developed to support this matching (e.g. the Damerau–Levenshtein distance, Bard (2007), based on the minimum edit distance by Levenshtein (1966)) , n-grams (Owolabi and McGregor, 1988), soundex (Odell, 1956) to name just a few, and which were used in the context of BiOnym. In this section we give an overview of methods that apply such techniques to taxon names matching.

Within the domain of taxonomic names/biological nomenclature, a considerable amount of work has been invested by the international biodiversity community in the creation of the Global Names Architecture (GNA) (GNA, 2014),

much of it supported by the Global Biodiversity Information Facility (GBIF) (Edwards et al., 2000) and its ECAT programme (GBIF, 2014), and by the Encyclopedia of Life (Wilson, 2003). The GNA has compiled a database of taxonomic names and its variants: the Global Names Index (GNI) (Patterson et al., 2010), which stands at nearly 20 million name strings. The GNI has a search interface (GNI, 2014a), but as far as we are aware, only allows for wildcard searching, and does not suggest similar names when searched with a non-matching name.

One of the tools developed under the umbrella of the GNA is a parser (GNI, 2014b), which can be used to split a taxonomic name in its individual components. The "GNI Parser", developed as a Ruby gem by Dmitry Mozzherin, is a component in several of the name matching services/applications discussed below. A description of the parser can be found in Boyle et al. (2013).

Taxamatch (Rees, 2008b), an algorithm that includes "fuzzy" name matching, is the basis for many applications of taxon name matching used by many biodiversity informatics systems (PESI, WoRMS, ALA, FishBase, etc.). The Taxamatch reference implementation is accessible through a web interface (IRMNG, 2014), and uses the Interim Register of Marine and Non-marine Genera as its Taxonomic Authority File. The interface allows for limited settings, most importantly to limit the TAF to a subset of IRMNG.

Several other implementations of Taxamatch have been created (Rees, 2014). The WoRMS Taxon match (WoRMS, 2014), uses a TAF specific for marine species; it includes a PHP/MS SQL port of Taxamatch, and uses the GNI Parser to split a name into its components.

In FishBase, there are two matching names systems installed (that do not analyse the species name authority): one searching for one name entered interactively by the user[2], and one that analyses a list copy-pasted in a box (under the section Tools as Match names[3]). Up to 2011, the entry search name used a simple matching algorithm that looked first for the full match, then the names matching the first three and last three letters of the entered name only, then the first two and last two letters, then either the full genus or the full species. It was a simple approach along the general philosophy of FishBase. Then the Taxamatch algorithm (Rees, 2008b) was adapted and implemented. The search consists of four steps: the full match, the Taxamatch, Genus or species full match, match of the first two and last two letters. A variation on this matching protocol was ported to Java for inclusion in BiOnym, and will be referred to as the "GSAy" algorithm ("Genus-Species-Authority (year)").

The Taxonomic Name Resolution Service (Boyle et al., 2013) builds on existing applications, including a PHP/MySQL port of Taxamatch, and the GNI Parser. Names can be standardised against several TAFs, at the time of writing all botanical. Though the focus of TNRS is on botany, its underlying design

[2] the general search page in www.fishbase.org

[3] http://www.fishbase.org/tools/upload/checkname.php

can be expanded. The TNRS is accessible through a web interface[4], and as a REST service. The matching can be fine-tuned by setting the required match accuracy.

The approaches described so far, see the matching process as a workflow. A general approach to build such systems uses Workflow Management Systems (WMSs). A WMS strongly separates the algorithms processing from the interaction among such algorithms. WMSs allow users with basic programming experience to combine algorithms and perform complex analyses. Usually, these systems rely on a common web area where the algorithms are published according to supported protocols, e.g. OGC WPS (Lanig et al., 2008). Algorithms are the atomic steps of the workflows and cannot be altered by WMSs users. Definitions of input and output types and other metadata allow users to understand and reuse algorithms developed by other users. Examples of WMSs are Taverna (Oinn et al., 2006), used in the European Project BioVEL (BioVEL Consortium, 2014), and Galaxy (Goecks et al., 2010), used in the biomedical and computational biology domain.

The BiOnym approach strongly emphasizes decoupling taxa names matchers, allowing to change the order of the matchers when building a matching workflow. Such workflow is assumed to be nested in a controlled environment, surrounded by pre-processing and post-processing phases.

Many other tools, online or off-line, exist to assist with taxon matching. Examples are Taxonome (Kluyver and Osborne, 2013); the Global Biotic Interactions programme (Global Biotic Interactions, 2014), which includes a taxon name matching component and is supported by Encyclopedia of Life (Wilson, 2003); the R package Taxize (Chamberlain and Szöcs, 2013); Taxonomic Nomenclature Checker (Taxonomic Nomenclature Checker, 2014); the taxon name parser of the Botanical Society of Britain and Ireland (Botanical Society of Britain and Ireland, 2014). Several issues, and functional taxon matching tools, have been posted by Rod Page on his blog site (Page, 2014). In this post, tools are provided to use Google Refine to match a set of taxon names with one or several reference files; reference files listed are EOL, NCBI, uBio, WoRMS, GBIF and GNI.

Given the number of taxonomic name matching systems, one might doubt the wisdom to produce yet another such system. But as noted before, the existing systems are often very rigid, and offer a solution within one particular context only, often tied to a particular taxonomic group, and with one use-case in mind. We set out to create a more generic system, by implementing the matching process as a flexible workflow, where many of the tuning parameters are under the control of the end user.

[4] http://tnrs.iplantcollaborative.org

3. The BiOnym Approach

The matching process follows a workflow approach, starting with a pre-processing step, followed by series of operators to do the actual matching, concluding with a post-processing step. The pre-processing includes a parser, to split a taxonomic name in its atomized components (e.g. splitting the string in the name proper and the authority field), and a resolver to settle common spelling variations (e.g. replacing all occurrences of "var." to "v."). The post-processing step defines how the results of the matching process are presented to the user. The matching itself is performed through a chain of atomic "matchers", where the output of each matcher is passed on as input for the next matcher in the chain. Each matcher decides, on the basis of customizable criteria, whether a pair of names should be considered as "matches", and splits the input list in "matched" and "non-matched" names. The matches go, with the criteria that were used to establish the match, to post-processing; the non-matched names are sent to the next matcher. Two broad categories of matchers are considered. A first type uses some kind of distance, such as the Levenshtein or Soundex distance. Another type of matcher, inspired by the "fuzzy" matching approach, applies a transformation to both test and reference names (e.g. strip off gender-specific suffix of specific epithet, or "stemming"), and then looks for matches. The matchers are configurable and it is possible to upload customized character/string substitutions to configure the pre-processing step and transformations used by the matchers.

BiOnym implements the approach described above, and is distributed as a standalone open source software written in the Java language[5]. One running instance of BiOnym has been integrated with the iMarine e-Infrastructure (Candela et al., 2009)[6], but the process is general enough to be separated from this system. By using iMarine, BiOnym benefits from Cloud computing, sharing and storage facilities. In particular, Cloud computing facilities (Candela et al., 2013) are able to process the user's entries in parallel fashion: the input list of species names is split into subsets and each subset is processed by one machine in the e-Infrastructure. Social networking and sharing facilities are used to download and analyse the output or to share it with colleagues. Although not indispensable for out process, these facilities enhance the potential of our workflow. On the other hand, also other taxa matching systems could benefit from the same facilities by following integration guidelines (Coro and Italiano, 2012). We give further details about the advantages and the modalities of integrating parallelisable algorithms in iMarine in the paper by Coro et al. (2014). Another advantage of using the BiOnym instance on iMarine, is that access policies to authority files are managed by the e-Infrastructure agreements with

[5] Available at https://svn.research-infrastructures.eu/public/d4science/gcube/trunk/data-analysis/EcologicalEngineSmartExecutor/src/main/java/org/gcube/dataanalysis/executor/nodes/transducers/bionym/

[6] Web interfaces are available for use, after authentication, at https://i-marine.d4science.org/group/bionym/bionym-app and at https://i-marine.d4science.org/group/bionym/taxa-names-processing

the data providers. The iMarine e-Infrastructure adopts users' privileges control and Virtual Research Environments mechanisms to manage visibility and accessibility policies for users. In the Virtual Research Environment created on the iMarine infrastructure for BiOnym, users are only presented access to authority files for which they have the necessary privileges.

To better explain the usage of the BiOnym workflow by a final user, we report a practical example. This use case reproduces a possible interaction between a user and the workflow.

The example use case is the following:

1. A set of scientific names is provided by a user as a raw CSV file, through a web interface (or directly when using BiOnym as a standalone process);
2. The file is transformed into a table, containing pre-formatted entries, to be used by the BiOnym workflow;
3. The user wants to check the spelling of the scientific names in the file, possibly getting correct transcriptions for them. "Correct" spelling is to be interpreted as corresponding with the TAF;
4. The user selects an authoritative collection of taxa names to use (e.g. FishBase), either among a finite choice proposed by the iMarine e-Infrastructure (when using the instance on iMarine) or indicated by the user as an http link (when using the standalone version). In particular, the user can provide a new authoritative collection under Darwin Core Archive format (Wieczorek et al., 2012). (S)he can also indicate several of these lists;
5. The user configures the BiOnym workflow in terms of (i) parsing procedures, (ii) matching mechanisms to use, (iii) lexical similarity tolerance parameters, (iv) accuracy of the recognition;
6. The system applies pre-parsing, parsing and post-parsing processing. Then, it applies a chain of matchers to the user's inputs, against the selected reference taxa names collection;
7. The system produces a table containing possible correct transcriptions for the entries, along with score indication and other metadata;

3.1. Taxon Names Parsing

The parsing of unstructured input data is a fundamental process that needs to be performed as a preliminary step, before the actual matching identification can take place. Its purpose is to clearly identify, out of the unstructured input data, the components relevant to the matching itself (the taxonomic atoms plus the authority and year). Conversely, the Taxonomic Authority Files are mostly available in Darwin Core Archive format. Thus, they clearly specify each taxonomic name part and need not to be parsed.

The input data parsing in the BiOnym workflow is a separate step with respect to the matching, and encompasses optional pre- and post-parse transformations. BiOnym pre-processes raw inputs before actually attempting to parse their structure, and processes the parsed results to enhance the quality of the identified atoms. Pre-parse processing mainly consists of a step to sanitise the input strings, e.g. removing question marks indicating uncertainty of the

identification; removing "sp. nov." and other qualifiers; making infra-specific indicators uniform by replacing all occurrences of "var." by "v.", etc. Post-parse processing assumes that the input string is correctly split into its atoms, and attempts to improve the quality of the input using this new information (e.g. checking and correcting capitalisation). Most of this pre- and post-parse processing is accomplished through regular expressions (with proper substitutions).

Currently, BiOnym can invoke two different parsers: the GNI Parser (introduced in Section 2), and a parser embedded in the BiOnym software (REGEXP or SIMPLE parser, Fiorellato (2015)). REGEXP provides good balance between speed and effectiveness: it is a heuristic parser relying principally on regular expressions. It applies a chain of substitution rules, combined with lexical similarities calculations, to extract the genus, species and authorship indications from a species scientific name. In our experiments, we noticed that the two currently available parsers (GNI and REGEXP) have complementary behaviour, as discussed in Section 5.

The parsing of input data is a relevant step of the matching process and the quality of the results achieved by the parser is crucial to the complete matching workflow of BiOnym. The BiOnym software architecture can be expanded with additional parsers whose implementation could be arbitrary, but must be wrapped in Java programming language. The technical documentation (Coro and Italiano, 2012) provides specifications for the input and output formats which these parsers must be compliant with.

3.2. Matchers

The BiOnym matchers are procedures that calculate similarity scores between 0 and 1: the score is calculated as the result of the comparison between the two taxon name records representing a user's input record and a reference record from a TAF. It is based on matcher-specific comparison logic.

Each matcher can access the full content of the two records being compared (from kingdom to species and authorities, including year of description) and can also take advantage of precomputed lexical indices contained in the TAFs, including the simplified atom forms, their soundexes and trigrams, as well as the genus and species stemmed versions.

Being $R = r_0, r_1, , r_N$ the set of reference data assumed as targets of the matching process and $I = i_0, i_1, , i_M$ the set of input data assumed as sources of the matching process, every BiOnym matcher applies the algorithm in figure 1.

Each matcher currently available in BiOnym implements the matching function $decimal :: matcher.match(\{property\}_i, \{property\}_r)$ in a different way. There are 6 matchers currently available in BiOnym, which we report into two categories:

Purely lexical oriented:

- Levenshtein: a weighted combination of the relative lexicographic Levenshtein distances (Bard, 2007), calculated between the scientific names and the authorities of an input and a reference entry. A weight is also applied to the relative lexicographic distance between the reported years,

if available. The weights sum to 1 and are by default set to 0.57 for scientific names, to 0.29 for authorities and to 0.14 for years distances. These values were chosen on the basis of heuristic evaluations on our test sets, but they can be modified by a developer when integrating the matcher in the BiOnym workflow;

- Soundex: adopts the same weighted combination as the Levenshtein matcher, but uses the relative Levenshtein distances of the soundex transcriptions (Odell, 1956) of the scientific names and of the authorities. The default weights are the same as the ones of the Levenshtein matcher.

- Trigram: calculates the relative size of the intersection between the trigrams (Owolabi and McGregor, 1988) of the scientific names and of the authorities separately. For the years, the relative Levenshtein distance is used. Weights are used to combine the scores and the default ones are the same as the ones of the Levenshtein matcher.

- Levenshtein + Soundex + Trigram: this matcher combines the scores from each matcher for scientific names, authorities and years distances separately. The combination is a weighted average of the scores. The weights are uniform by default, but can be modified by a developer when integrating the matcher in the BiOnym workflow.

Taxonomic names oriented:

- GSAy: the "Genus - Species - Authority - year" matcher differs from the matchers described above, in the sense that it is specific for taxonomic names. The different words constituting the user's scientific name are compared one by one with the words of the reference name. Matching is first performed on the original words; afterwards comparisons are done on words which have been "stemmed" (i.e. without suffix) and cleaned of non influential characters. Different weights can be assigned to the matching of different words: e.g., a difference in the year of description can contribute less to the final distance score, than a difference in the genus or species name. The default configuration assigns equal weights to all the words.

- Taxamatch: the matcher developed by Rees (2008b) and cited in Section 2.

Both Taxamatch and GSAy were originally developed outside BiOnym, and ported to Java to make them available for BiOnym. These matchers are specific for matching of strings that are representations of taxonomic names. Both incorporate knowledge of taxonomic literature. This sets them apart from the four purely lexical matchers described above.

3.3. The Matching Process

BiOnym implements a chained process in which several matchers are called in sequence. The first step is always the application of a complete scientific name

parser. As explained in the previous sections, the aim of this step is to split the input string into a species scientific name, followed by an authorship indication. Furthermore, the scientific name is possibly divided into genus, subgenus and species name indications. This step should help the next sequence of matchers in recognizing the correct transcription for the input. One justification for the usage of a processes chain instead of a single matching step is also that the dependency on the input parser must be flexible. A strong dependency on the parser's output would in fact limit the performance of a matcher. Some of the matchers in the chain can be more influenced by an error in the parsing step (e.g. the Taxamatch matcher), while other ones can be more tolerant (e.g. the Levenshtein matcher). This flexible behaviour by the chain is more evident when analysing the performance on species names that contain errors that are uniformly distributed along the string.

Figure 2 depicts one possible matching chain, which starts with the application of the REGEXP parser and then applies a chain of matchers. In the sequence, highest priority is given to the entries that are recognized by those matchers who come first. Each matcher produces a list of possible transcriptions for the parsed input string. On the other hand, if a transcription has been recognized by a matcher at a previous step, with whatever score, it will not be overwritten. The transcription, along with the score, remains the one recognized by the previous matcher, even if the score given by the later matcher is higher. In other words, the matchers which come later will "trust" the previous ones. Two transcriptions are assumed to be equivalent if they present the same complete scientific name, the same author and year indications and the same identification code according to the reference dataset. This is equivalent to pass, at each step, only those names that have not been recognized by the previous matcher. An alternative approach would have been to collect the scores from the matchers and merge them in the end. Implementing this solution requires combining heterogeneous matchers, this would introduce subjectivity when comparing scores from different matchers, scores that are incommensurable and so not simply comparable. Moreover, none of the (subjective) approaches we tried resulted in a better final result than the hierarchical approach.

Thus, the matching chain we propose adopts an enrichment approach for the list of matching names. Every matcher is allowed to produce a list of spelling variations which enriches the previous one, without overwriting the list of variations already found. A possible drawback of such an approach is that, if a previous matcher recognizes the correct reference entry with a low score, none of the other matchers will raise this score, thus the correct answer will not be at the top of the list. This phenomenon has the effect to increase the recall of the system, but to lower the precision at the same time. In order to alleviate this problem, it is wise to put less effective but highly precise matchers at the first positions in the chain.

The matching chain allows any combination of matchers during the configuration phase. Figure 3 depicts the web user interface we implemented, allowing

users to configure the complete chain[7]. At the start, the user can chose to activate or to disable pre-parse processing. The drop down menu allows choosing the reference dataset against which the process applies the matchers. Then (s)he can choose the sequence of matchers, the maximum length of the list produced by each matcher and the recognition threshold for each matcher score, under which the match will be not reported. At the end of the process, the system returns a table with the list of possible matches for the input string. This list can contain a maximum number of entries corresponding to the sum of the list lengths allowed for the matchers. This is the theoretical case of a sequence of matchers that were absolutely complementary. An example of output of on the entry "Gadus morrhua (Linnaeus, 1758)" is reported in Table 2. The score is given in decreasing order and the correct transcription is reported having the scientific name separated from the authority. Furthermore, the identification code of the matching name in the reference source is reported for each entry.

The BiOnym matching process is optimized when the first matchers in the chain are those which incorporate expert knowledge. Thus, the matches provided by these should have higher priority in the chain than purely lexicographic matchers. We will show in Section 5 that such approach generates a matching chain that commits complementary errors with respect to the single matchers. Based on the performance evaluation and on the complementarity of the errors by the matchers, we set up a default matching chain for those biologists that are more interested into biodiversity studies. We supposed these scholars could be more interested in (i) having different performance with respect to a pure lexicographic approach, and (ii) producing output that involved expert knowledge embedded in the matchers. On the other hand, they would accept to have a wider spectrum of proposals with valid alternatives. This implies to sacrifice the precision of the system with the advantage to have a wider choice of possible complementary alternatives (higher recall). This approach does not fit the requirements of fishery managers, who are usually more interested in having precise results in short time. This happens because they usually manage large quantities of data to be checked.

As biodiversity-oriented matching chain, we propose just the process depicted in Figure 2, in which the GSAy matcher and Taxamatch appear at the first places because they are based on expert knowledge, thus supposed to be more strict in recognizing but also more precise. As application for fishery managers, we would suggest a short and fast matching chain that applies a REGEXP parsing step, followed by one Levenshtein matcher. In Section 5 we will justify such choice based on the performance reports.

3.4. Post processing

Once the matching process is terminated, results have to be processed; what exactly this post-processing entails is dependent on the particular use case. In

[7] Available, after authentication, at https://i-marine.d4science.org/group/bionym/bionym-app

many cases, where the end-user is trying to clean up an input list of names by comparing it to a reference list, the post-processing might consist of presenting a list of alternative names from the reference list for each of the names of the input list. Another possible post-processing step might be to follow synonym links in the TAF, to replace a (possibly originally misspelled) synonymous name with the currently valid name.

BiOnym offers opportunities to customise the settings of the matching process, so that end-users can look for the best settings corresponding to their matching needs. This flexibility causes the need for a system to evaluate performance of alternative settings. While developing BiOnym we built a system to evaluate alternatives using trigraphs and AUC curves; the precise metrics we used in this evaluation are the ones presented in Section 5. The evaluation system was built with the R programming language and is described in detail in Vanden Berghe et al. (2014). The R code is available from a public SVN code repository[8].

4. Data

In this section we describe the format of the reference datasets used by our process and the test datasets we prepared to evaluate the process.

4.1. Reference Files: the Taxonomic Authority Files

BiOnym offers the user a choice of several Taxonomic Authority Files (TAFs), including the option to provide his/her own list. When using the instance on the iMarine e-Infrastructure, internationally recognized references are dynamically linked to the e-Infrastructure resources; this avoids issues with intellectual property rights, and eliminates the inconvenience of keeping the TAFs up to date on the iMarine infrastructure. The following lists were available in the e-Infrastructure at the time of writing: Catalogue of Life, FishBase, World Register of Marine Species, Interim Register of Marine and Non-marine Genera, National Center for Biotechnology Information, and the Integrated Taxonomic Information System.

We used taxonomic tables containing lists from the Ocean Biogeographic Information System (OBIS) contributors (Berghe et al., 2010) as authority files to test our process. This was done for practical reasons, and in no way implies that BiOnym would want to promote these working tables as real taxonomic reference files. Next section explains our rationale to select these tables as TAFs and as test sets. In much of our testing, we used FishBase as the authority file. As of May 2014, it contains about 87,000 names of which about 9,200 are misspellings that may substantially decrease the number of false positive. It contains information on names (type of synonymy, misspelling, etc.) that may help to refine the post-processing step, including links to Catalog of Fishes (Reis, 2000) for on-the-fly checking of latest validity assessment.

[8] https://svn.d4science.research-infrastructures.eu/

The Taxonomic Authority Files (TAFs) to provide to BiOnym should follow a format based on the Darwin Core Archive (DwCA) specifications (Wieczorek et al., 2012). This TAF Format includes results of several calculations, such as stemming, to speed up string matching processes. The TAF files can also store standard taxonomic classification or to vernacular names. When taxa and vernacular names come from the same sources, they are linked with cross reference identifiers.

The stored TAFs can be accessed by the BiOnym workflow via multiple protocols including HTTP(S), FTP(S) or Java classpath. Data from the TAFs are streamed upon request, thus they are not kept in memory for the full duration of the matching process. This ensures that BiOnym will have a memory footprint suited to the processing machine.

TAFs are stored as compressed CSV files. The structure of TAFs includes all the taxonomic ranks (from kingdom to infraspecies), and taxonomic authority, for each entry. The entry has a unique ID, which is possibly the one reported by the original data source provider (e.g. WoRMS:300760). Furthermore, for each entry, a TAF reports the following pre-computed information:

1. a simplified version of the atom. E.g. the entry is reported in ASCII, uppercase with non-letter characters removed;
2. the full sequence of trigrams (Owolabi and McGregor, 1988);
3. the soundex transcription, also for the entire scientific name;
4. the stemmed version, with double letters replaced by single ones and Latin suffixes removed. The stemming phase applies to genus and species atoms only.

As example, we report the TAF line for *Latimeria chalumnæ*:

1. Simplified version: LATIMERIA CHALUMNAE. Here, also the original æis transformed into AE;
2. Trigrams: LA LAT ATI TIM IME MER ERI RIA IA CH CHA HAL ALU LUM UMN MNA NAE AE;
3. Soundex: L356 (for "LATIMERIA"), C450 (for "CHALUMNAE"), L356245 (for "LATIMERIA CHALUMNAE")

Stemming genus and species is accomplished by removing double letters and common suffixes from the original, simplified strings. Thus, the stemmed versions of Gadus (genus) and mediterraneus (species) would be GAD for the genus and MEDITERANE for the species name, where the double R was removed too. The same set of pre-computed values, without the stemming step, is produced also in vernacular names TAFs.

As the process of stemming genus and species is potentially relevant for multiple distinct matchers, it is accomplished at TAF generation level rather than at runtime. Furthermore, the double-letter removal at stem level flattens the differences that can be encountered within different spellings of the same scientific name, both for input and for reference data. Still, also when not

using stemmed genus and species, double letters could be removed during the preprocessing phase but this will have an impact on input data only.

BiOnym includes a standalone command line tool that can produce TAF files out of any valid DwCA file. Currently, TAF files are not shipped with the BiOnym software distribution. Some of them are available in the iMarine e-Infrastructure after the original data providers have established access and citation policies. Alternatively, users can produce their own TAF files directly, assuming they have access to the DwCAs of interest; this also facilitates building TAFs with the right taxonomic scope, which is important in avoiding false positives.

4.2. Test Data

In order to check the performance of BiOnym, we needed test input lists of names, for which the correct spelling was known. For this, we used a double approach. Part of our testing was done on the basis of OBIS data, using OBIS quality-controlled names as TAF, and the names as submitted by OBIS providers as test names. In what follows the latter are referred to as "real misspellings". In a second approach we introduced character substitutions in a list of known-good names. We will refer to these misspellings as "simulated misspellings".

To generate input lists of real misspellings, we took samples of names from the Ocean Biogeographic Information System database. Taxonomic names, as found in the submissions of contributors of OBIS biogeographic information, have been curated, and manually merged into the table with taxonomic names already present in OBIS. We refer to these as "harmonised names". The first goal of this process is to harmonise the spelling, and create a consistent list of names. The consequence, as regards BiOnym, was that we had many name strings, as they circulated "in the wild", that were manually matched with a reference list with consistent, harmonised spelling. This allowed us to set up experiments where the "correct spelling" was known, and so to discriminate between true and false positives. In the 2011 version of the OBIS data we used, there were 424,500 different name string records, corresponding to 202,726 harmonised names.

We took complete scientific names, verbatim as coming from OBIS data contributors, and selected those that were different from the correct name string. There are several reasons why name strings would be different from the standard form. Apart from straightforward typos and other spelling variations, often the name field included information that was supposed to be in other fields, such as information on gender, life stage or reliability of the identification of the specimen. We knew the correct version of the name, as this was determined in the process of curating the OBIS data. We were also able to restrict these real misspellings to names of Pisces, through OBIS' link with the taxonomy. We used 32 sets containing 1024 taxa names with associated authorship to evaluate the performance of our system, both against WoRMS and against FishBase (on which the fish part of WoRMS is based). We also sent the same names to the Taxamatch process hosted by WoRMS.

For the lists of simulated misspellings, we produced artificial misspelling errors by implementing a procedure, in the R programming language, that introduced lexical errors in the scientific names. It produced random alterations of string characters at random positions; R code is open source and publicly available[9]. In other words, it simulates random noise in the string transcription. The noise is introduced at places they are likely to occur, e.g. in the declension. As for the real misspellings, we generated 32 sets of 1024 misspelled names.

Examples of real misspellings are *Abudefduf sp*, *Abudefduf dicki* (instead of *Abudefduf dickii*). Examples of simulated misspellings are *Abramis micuopteryx* Dalenciennes, 1845 (instead of *Abramis micropteryx* Valenciennes, 1844), *Aetomylaeus hulepti* Smitm, 1961 (instead of *Aetomylaeus huletti* Smith, 1953). For both real and simulated misspellings, the files used in the analysis are publicly available[10].

5. Results

In this section, we report the performance of our system at several levels. We compare the effectiveness of two parsing procedures, one of which is a state-of-the-art expert system for scientific names used both by the WoRMS (Costello et al., 2013) and the IRMNG (Rees, 2008a) data providers. On the other hand, we report the performance of each matcher involved in the workflow we configured for biodiversity scholars (see Figure 2). Finally, we investigate the degree of complementarity of simple workflows with respect to a complete workflow and the efficiency of the process.

5.1. Evaluation Metrics

In order to evaluate the parsers in terms of accuracy and processing speed, we used the test datasets described in Section 4.2. Accuracy is measured as the fraction of correctly parsed names with respect to the total number of names provided to the system. Human expert opinion was used to verify the cases of disagreement between the two parsers (GNI and REGEXP). Evaluation was made both on the effort for distinguishing genus from species names and for separating author names from years.

We used the test datasets described in Section 4.2 also for evaluating the complete workflow. On these benchmarks, we measured standard quantities used in the evaluation of Information Retrieval systems (Harman, 2011; Wikipedia, 2015). In particular, we averaged the following quantities on the 32 sets for each type of input:

[9] At this link, provided by a CNR high-availability distributed storage system http://goo.gl/WoPJOH

[10] At this link, provided by a CNR high-availability distributed storage system http://goo.gl/5Fcwu0

$$RecognitionPercentage = \frac{TruePositives}{Total\,Number\,of\,Input\,Species\,Names}$$

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}$$

$$Fmeasure = 2 * \frac{Precision * Recall}{Precision + Recall}$$

We report such quantities in the evaluation tables, to assess the differences in the quality of the systems. Finally, we evaluated the efficiency of our system, at the level of the overall computation time of the entire workflow as well as of the matchers. We show the benefits coming from the usage of Cloud computing and the pros and cons of using smart search strategies for lexical similarities.

5.2. Parsing

In order to evaluate the differences in the accuracy of two parsers currently available in BiOnym (GNI and REGEXP), we tested their behaviour on a set of 1023 real misspelled names. We focused on the discrepancies between results produced by each parser. Whenever the two parsers identified different atoms for a given input data, we let a human expert decide which of the two parsers had returned the correct result. A comparison is reported in Table 3-a, where the parsers were asked to distinguish between genus and species only. The agreement between the GNI and REGEXP parsers was 90.32%.

To better investigate disagreement cases, we used human experts' evaluation on the 99 differently parsed results. The result is reported in Table 3-b: 53.53% of the results from REGEXP were correct, while GNI correctly parsed only the 5.05% of the entries. Additionally, 55.55% of the GNI results were confirmed to be wrong (9.09% for REGEXP) and both the parsers produced a number of uncertain results ranging from 37.37% (REGEXP) to 39.39% (GNI).

A comparison on the parsing of authorities and authority years is reported in Table 3-c. Also in this case the 76 disagreement cases were evaluated by a human expert (Table 3-d). The GNI and REGEXP parsers produced parsed authorities and years that were identical in 92.57% of the cases. For the remaining 7.43%, the REGEXP produced 93.42% of correct results while GNI only 2.63%. Additionally, 96.05% of the GNI results for differently parsed entries where confirmed as wrong (3.94% in the case of REGEXP) and both parsers produced a number of uncertain results ranging from 1.31% (GNI) to 2.63% (REGEXP).

Table 4 gives a qualitative comparison of the parsers used by BiOnym. We associated a quality indication to three aspects we consider crucial in these systems. In terms of computational efficiency, the GNI parser took almost 65 seconds to parse scientific names and authorities for 1023 entries. The scientific names were submitted in batches of 100 entries per request, which resulted in

63.13 ms per entry. On the other hand, the REGEXP parser took 219 ms (0.21 ms per entry), with the 1023 entries submitted as a single batch. Thus, REGEXP resulted to be about 295 times faster than GNI. Times were measured on an i5-3470 CPU @ 3.20 GHz with 8.00 GB of RAM, running Windows 7 and with a broadband Internet connection measured as A+ both on National and Global Grades via Ookla Speedtest[11]. From the point of view of parsing "resilience" to noise, GNI has the disadvantage to be strictly related to the expert rules it embeds. Thus, it is more sensitive to noise with respect to the REGEXP parser, which is based on algorithms commonly used for syntax correction and on general parsing rules. For what regards accuracy, interpreted as the correct parsing rate of non-noisy data, GNI is better than REGEXP, because it can parse also taxonomic names over the genus rank. Furthermore, the GNI expert rules are specifically designed to manage taxonomic names notations. The input data we used for this experiment are publicly available[12]. Also GNI and REGEXP parsed results for these same inputs are publicly provided[13].

We experienced that the quality of the input parsers depends on the application context. Our impression is that the GNI Parser is more suited to parse raw input data that are complex and well structured, correct capitalization for both scientific names and authorships. On the other hand, the REGEXP parser is more robust, possibly better suited when short parsing time is required. Usually, this is the case when a large amount of input data must be parsed and the quality of the input names is relatively low. Indeed, we noticed that sub-optimal results can be tolerated when dealing with extremely noisy input data.

5.3. Matchers

Table 5-a reports the performance of several simple workflows made up of one REGEXP parser step, followed by only one matcher. The performance is reported on the "real misspelling" names recognition. From Table 5-a it is notable that the recognition percentage decreases when the maximum number of suggested transcriptions increases. In the same way, the F measure increases for almost all the matchers when the output list is shorter. This increment in the F measure is due to the increase of the *Precision* and to the fact that the *Recall* remains high even if the output list is shorter. The best matcher on real names is the Trigram-based workflow. Nevertheless, the GSAy-based is the one losing the least amount of recognition percentage when the number of outputs changes.

Table 5-b reports the performance of the simple workflows on the simulated misspellings. With high degree of noise Levenshtein performs much better than the others. More rigid matchers, like GSAy and Taxamatch, gain lower performance. The Trigram-based workflow suffers from the fact that errors are

[11] Interface available at http://www.speedtest.net

[12] At this link, provided by a CNR high-availability distributed storage system http://goo.gl/dlXaC0

[13] At this link, provided by a CNR high-availability distributed storage system http://goo.gl/0l1NV4 and http://goo.gl/anqxw5

uniformly distributed along the string, thus all the trigrams are likely to contain errors. The good performance of Levenshtein distance stems from the fact that the calculated distance is not influenced by the position of the discrepancies between the two strings.

5.4. Workflow

In order to assess the quality of BiOnym, we compared the performance of several configurations of the workflow. In particular, as benchmark system we used a workflow that invoked either the REGEXP or the GNI parser, followed by:

1. The GSAy matcher, with recognition threshold equal to 0.6 and 10 maximum allowed transcriptions;
2. The Taxamatch algorithm, with recognition threshold equal to 0.6 and 10 maximum allowed transcriptions;
3. The Levenshtein matcher, with recognition threshold equal to 0.4 and 10 maximum allowed transcriptions;
4. The Trigram matcher, with recognition threshold equal to 0.4 and 10 maximum allowed transcriptions;

We chose this workflow on a heuristic basis. Tests aimed at building a sequence that exploited complementary information as far as possible. For such reason, we did not add the Soundex-based matchers, because these use a lexicographic approach that was very similar to the Levenshtein one.

We compared the BiOnym workflows with the Taxamatch-based system provided via Web Service by the WoRMS data provider. This system applies a parsing step, based on the GNI parser, followed by a set of expert rules and lexicographic distances. The comparison we report is consistent, because we used the datasets described in Section 4.2, which rely on the Pisces Class of WoRMS.

Tables 6-a and 6-b report the result of such comparison on real and simulated misspellings respectively. The BiOnym workflow uses the GNI parser and the REGEXP parsers alternatively. We report the performance as a function of the variation of the maximum number of allowed matches in the final output. Furthermore, in the tables we report the performance of the simple Levenshtein-based workflow used in the previous section. We chose this system because of its high performance, and because it is more stable when passing from real to artificial inputs. The recognition percentage of BiOnym decreases when the output length decreases. On the other hand, the F measure increases with decreasing the output length, which suggests that this configuration of the workflow would make BiOnym perform well when used as an information retrieval system.

The tables highlight that the WoRMS taxa matching algorithm performs very well on real misspellings, while it lacks of robustness on simulated misspellings. On the other hand, BiOnym gains higher recognition performance when the output length is 10, but it is worse than the Taxamatch-based system if viewed as an information retrieval system, because the F measure is lower.

The Levenshtein-based workflow gains higher recognition performance than the complete BiOnym and is comparable to the Taxamatch-based system in terms of F measure, when the output list length is fixed to 1. Even if the recognition percentage of BiOnym is generally lower than the Levenshtein-based workflow, its usage is justified by the complementary errors it is able to detect. Detecting such errors, in fact, is very important in biodiversity-oriented applications, despite some loss in performance.

5.5. Complementarity Analysis

Table 7 reports the percentages of complementary errors committed by the workflows. In particular, it reports the percentage of species recognized by the workflows on the left side of the table with respect to those at the upper side of the table. The workflows with the highest performance gain highest complementarity percentages. We report the comparison both on real misspellings and on simulated misspellings, in order to highlight the behaviours in completely different scenarios. The highest percentages are always recorded with respect to the GSAy matcher, which only uses expert rules. With the term "BiOnym workflow", we mean the one used in the evaluation of the previous section, made up of a sequence of matchers beginning with GSAy and ending with the Trigram matcher. Furthermore, for this experiment we allowed up to ten names in the output list and used the REGEXP parser. Although this workflow gains lower performance with respect to the Trigram and Levenshtein-based workflows, it still recognizes complementary species names. These cases can be very interesting from the point of view of a biodiversity researcher, but they could be a hindrance to those operators who must correct a huge amount of taxa names. Examples of complementary species names highlight the differences in the behaviour of the matchers: the Trigram-based workflow is able to recognize "*Gobio gobio saramaticus* Berg, 1949" as the correct name for the input "*Gobio gobio saramatwcxs* Berg, 1949", while the Levenshtein-based is not able to recognize it. This happens because the errors in the string are concentrated only in one of the trigrams and the string is quite long. Thus, the Levenshtein-based workflow is more influenced by the length of the string, while Trigram-based workflow only cares about how many trigrams the reference and the input strings have in common. This is the main reason why the Trigram-based workflow gains the highest performance on the real misspellings.

Another example is the input "*Arvoglhssus thoro* Kyle, 1913", which GSAy correctly recognizes as "*Arnoglossus thori* Kyle, 1913", and that Taxamatch cannot recognize. This is due to the concentration of one of the errors in the declension of "thori", which is ignored by GSAy but taken into account by Taxamatch. GSAy compares only the stems of the words, thus it finds a correspondence between the species names and the authorships and gives a non-zero score. On the other hand, Taxamatch detects mismatches between the genus names and between the species names, thus it discards it as a match.

5.6. Efficiency

A comparison between the efficiency of the parsers has already been reported in Section 5.2, where we highlighted that using the GNI parser may enhance accuracy in some contexts at the expense of efficiency. For what regards the workflows, we compared efficiency as the total execution time when invoked by a remote thin client. On one side, we setup a web client that interrogated the BiOnym workflow instance residing on an iMarine Web Server (Candela et al., 2013). The BiOnym workflow used in this comparison is the default one described in Section 3.3 and used in Section 5.4. Another web client invoked a remote WoRMS Web Service hosting the Taxamatch algorithm we have taken as reference so far. For this comparison, we used WoRMS as TAF. In the evaluation, we calculated the average time required to produce results for 1024 species names containing real misspellings. The WoRMS service could be invoked for 50 species at time, while BiOnym could be invoked on the 1024 species directly. In the process, BiOnym applied four matchers sequentially, while the WoRMS Taxamatch was an all-in-one procedure. The iMarine e-Infrastructure service hosting BiOnym was able to parallelize the execution on 21 machines running CentOS 5.7 x86 64 operating systems, with 2 CPUs, 2 GB of RAM and 10 GB of disk space. Each machine received a set of 50 species at time to be processed using BiOnym.

In the end, the average recorded time was **7.3 minutes** for BiOnym and **19.6 minutes** for Taxamatch. A sequential run of BiOnym would have required much more time but the benefits of the parallelisation emerge. The shorter execution time for BiOnym justifies the usage of Cloud computing. This allows building long chains of matchers and exploiting the complementary behaviour of the matchers, which also results in higher accuracy.

6. Discussion

In the introduction to this paper, we have stressed the importance of names as identifiers, facilitating the integration of information from different sources. Obviously, in order to play this role, these names have to be kept as "clean" as possible to recover data, information and knowledge about species (or taxa in general), and to be able to find natural resource management recommendations and regulations for local (e.g., protected area) and/or global levels (international conventions on biodiversity: CBD, CITES, RAMSAR, etc.).

Generic search engines like Google are already doing a great job with lexical tools. But they fail to find the spelling variants stored in online systems such as WoRMS. Dedicated tools like those implemented using Taxamatch (WoRMS, PESI, GNA/GNI, CoL, etc.) and BiOnym are much more efficient in their - much narrower - application domain. There are certainly synergies to be looked for and developed to further implement the semantic web.

Beside databases, names are also conveyed through articles in scientific journals, faunas and floras, field guides, etc. We may regret that scientific journal editors in general did not integrate in their reviews the correction of scientific

names. It is clear that 30 years ago, informatics tools were not developed or made available as they are today. Although some progress was made (e.g., PenSoft Publ.), journals that do not deal with biodiversity directly but rather do (molecular) biology in general are still ignoring that important aspect up to the point that scientific names are not used at all. This is notable from several journal guidelines that indicate heterogeneous usages of scientific names (e.g., Biological Abstracts (2015); Bragantia (2015); Journal of Nematology (2015)), that sometimes even suggest using common names instead of scientific names. Tools such as Taxamatch and BiOnym will be able to assist authors and editors by offering facilities to harmonise spelling of taxonomic names, and to prevent misspelled names from polluting the literature.

In addition, these tools could also be used for searching the literature by generating possible (reasonable) spelling variants to be matched, with the purpose of query expansion. It can be done automatically like in the case of the letter simplification in the fuzzy matcher (Taxamatch), or by customising expert rules: e.g., if the user searches for "longirostris", then search also for "longinasus", "longirhynchus" and all the possible variations of "rhynchus".

In this paper, we have demonstrated that BiOnym compares favourably with other systems, both in terms of efficiency (time needed to perform a comparison) and effectiveness (quality of the results of the comparison). This does not come as a surprise, since during the development of BiOnym, the achievements of the others were taken into consideration. The philosophy of BiOnym's development was to make it possible to incorporate efforts by others, managing related acknowledgment. BiOnym is meant to be an open framework for continuous development, not a monolithic, static, software tool. BiOnym is open for any scientist, and anyone can contribute with matchers or parsers, or can explore existing ones. This allows quantitatively comparing the performance of different matchers and their settings. Most importantly, using the BiOnym instance on iMarine allows researchers to concentrate on taxonomic name matching rather than on developing data access or processing facilities. Currently, BiOnym is being used by more than 20,000 users per month via the iMarine e-Infrastructure and we foresee that this number is going to increase.

Acknowledgments

The authors want to thank David (Paddy) Patterson, Tony Rees and Dmitry Mozzherin for the many discussions we had on taxon name matching, and especially Dmitry for his assistance with the GNI parser. The work reported has been mainly supported by the *iMarine* project (FP7 of the European Commission, FP7-INFRASTRUCTURES-2011-2, Contract No. 283644).

References

Bard, G. V., 2007. Spelling-error tolerant, order-independent pass-phrases via the damerau-levenshtein string-edit distance metric. In: Proceedings of the

fifth Australasian symposium on ACSW frontiers-Volume 68. Australian Computer Society, Inc., pp. 117–124.

Berghe, E. V., Stocks, K. I., Grassle, J. F., 2010. Data integration: The ocean biogeographic information system. Life in the World's Oceans: Diversity, Distribution, and Abundance, 333.

Biological Abstracts, 2015. Biological Abstracts data description guide. URL http://www.library.illinois.edu/bix/pdf/dbguide/bioabs.pdf

BioVEL Consortium, 2014. The BioVEL European Project. Http://www.biovel.eu/.

Bisby, F. A., 2000. The quiet revolution: biodiversity informatics and the internet. Science 289 (5488), 2309–2312.

Bisby, F. A., Froese, R., Ruggiero, M. A., Wilson, K. L., 2004. Species 2000 and ITIS catalogue of life, annual checklist 2004: indexing the world's known species. CD-ROM.

Botanical Society of Britain and Ireland, 2014. Taxon name parser. Http://bsbidb.org.uk/taxonnameparser.php.

Boyle, B., Hopkins, N., Lu, Z., Garay, J. A. R., Mozzherin, D., Rees, T., Matasci, N., Narro, M. L., Piel, W. H., Mckay, S. J., et al., 2013. The taxonomic name resolution service: an online tool for automated standardization of plant names. BMC bioinformatics 14 (1), 16.

Bragantia, 2015. Bragantia authors guidelines. URL http://www.scielo.br/revistas/brag/iinstruc.htm

Candela, L., Castelli, D., Coro, G., Pagano, P., Sinibaldi, F., 2013. Species distribution modeling in the cloud. Concurrency and Computation: Practice and Experience, n/a–n/a. URL http://dx.doi.org/10.1002/cpe.3030

Candela, L., Castelli, D., Pagano, P., 2009. D4science: an e-infrastructure for supporting virtual research environments. In: IRCDL. pp. 166–169.

Chamberlain, S. A., Szöcs, E., 2013. taxize: taxonomic search and retrieval in r. F1000Research 2.

Coro, G., Candela, L., Pagano, P., Italiano, A., Liccardo, L., 2014. Parallelizing the execution of native data mining algorithms for computational biology. Concurrency and Computation: Practice and Experience, n/a–n/a. URL http://dx.doi.org/10.1002/cpe.3435

Coro, G., Italiano, A., 2012. Statistical Manager developer's guide. Http://gcube.wiki.gcube–system.org/gcube/index.php/How–to_Implement_Algorithms_for_the_Statistical_Manager.

Costello, M. J., Bouchet, P., Boxshall, G., Fauchald, K., Gordon, D., Hoeksema, B. W., Poore, G. C., van Soest, R. W., Stöhr, S., Walter, T. C., et al., 2013. Global coordination and standardisation in marine biodiversity through the world register of marine species (worms) and related databases. PLoS One 8 (1), e51629.

Edwards, J. L., Lane, M. A., Nielsen, E. S., 2000. Interoperability of biodiversity databases: Biodiversity information on every desktop. Science 289 (5488), 2312–2314.

Fiorellato, F., 2015. The REGEXP Parser.
URL `http://wiki.i-marine.eu/index.php/YASMEEN_input_data_parser#SIMPLE_parser_processing_rules`

Froese, R., 1997. An algorithm for identifying misspellings and synonyms in lists of scientific names of fishes. Cybium 1 (3), 265–280.

Froese, R., Pauly, D., 2000. FishBase 2000: concepts, design and data sources. WorldFish, Jalan Batu Maung, Batu Maung, 11960 Bayan Lepas, Penang, Malaysia.

GBIF, 2014. The GBIF ECAT programme. Https://code.google.com/p/gbif-ecat.

Global Biotic Interactions, 2014. GloBI. Https://github.com/jhpoelen/eol-globi-data/wiki.

GNA, 2014. The Global Names Architecture. Http://www.globalnames.org/.

GNI, 2014a. The Global Names Infrastructure Interface. Http://gni.globalnames.org.

GNI, 2014b. The GNI parser. Http://gni.globalnames.org/parsers/new.

Goecks, J., Nekrutenko, A., Taylor, J., et al., 2010. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol 11 (8), R86.

Harman, D., 2011. Information retrieval evaluation. Synthesis Lectures on Information Concepts, Retrieval, and Services 3 (2), 1–119.

IRMNG, 2014. Taxamatch Web Interface. Http://www.cmar.csiro.au/datacentre/irmng/.

Journal of Nematology, 2015. Journal of Nematology guidelines.
URL `http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3700740/`

Kluyver, T. A., Osborne, C. P., 2013. Taxonome: a software package for linking biological species data. Ecology and evolution 3 (5), 1262–1265.

Lambe, P., 2014. Organising knowledge: taxonomies, knowledge and organisational effectiveness. Elsevier.

Lanig, S., Schilling, A., Stollberg, B., Zipf, A., 2008. Towards standards-based processing of digital elevation models for grid computing through web processing service (wps). Computational Science and Its Applications–ICCSA 2008, 191–203.

Levenshtein, V. I., 1966. Binary codes capable of correcting deletions, insertions, and reversals. In: Fortov, V. E. (Ed.), Soviet physics doklady. Vol. 10. MAIK Nauka, pp. 707–710.

Odell, M. K., 1956. The profit in records management. Systems Magazine 20–21.

Oinn, T., Greenwood, M., Addis, M., Alpdemir, M. N., Ferris, J., Glover, K., Goble, C., Goderis, A., Hull, D., Marvin, D., Li, P., Lord, P., Pocock, M. R., Senger, M., Stevens, R., Wipat, A., Wroe, C., 2006. Taverna: lessons in creating a workflow environment for the life sciences. Concurrency and Computation: Practice and Experience 18 (10), 1067–1100.
URL `http://dx.doi.org/10.1002/cpe.993`

Owolabi, O., McGregor, D., 1988. Fast approximate string matching. Software: Practice and Experience 18 (4), 387–393.

Page, R. D. M., 2014. iphylo. Http://iphylo.blogspot.be/2012/02/using-google-refine-and-taxonomic.html.

Patterson, D. J., 2014. Helping protists to find their place in a big data world. ACTA PROTOZOOLOGICA 53 (1), 115–128.

Patterson, D. J., Cooper, J., Kirk, P. M., Pyle, R., Remsen, D. P., 2010. Names are key to the big new biology. Trends in ecology & evolution 25 (12), 686–691.

Rees, T., 2008a. 18.8. irmng–the interim register of marine and nonmarine genera. The Proceedings of TDWG, 72.

Rees, T., 2008b. 8.3. taxamatch, a fuzzy matching algorithm for taxon names, and potential applications in taxonomic databases. The Proceedings of TDWG 35.

Rees, T., 2014. A collection of software for taxon names matching. Http://www.cmar.csiro.au/datacentre/taxamatch.htm.

Reis, R. E., 2000. Catalog of fishes. Copeia 2000 (3), 904–906.

Taxonomic Nomenclature Checker, 2014. TNC. Http://pgrdoc.bioversity.cgiar.org/taxcheck/.

Vanden Berghe, E., Bailly, N., Coro, G., Fiorellato, F., Aldemita, C., Ellenbroek, A., Pagano, P., 2014. Bionym: a flexible workflow approach to taxon name matching. Technical report, CNR, technical report, 2014.

Wieczorek, J., Bloom, D., Guralnick, R., Blum, S., Döring, M., Giovanni, R., Robertson, T., Vieglais, D., 2012. Darwin core: An evolving community-developed biodiversity data standard. PLoS One 7 (1), e29715.

Wikipedia, 2015. Information Retrieval, Wikipedia page.
URL http://en.wikipedia.org/wiki/Information_retrieval

Wilson, E. O., 2003. The encyclopedia of life. Trends in Ecology & Evolution 18 (2), 77–80.

WoRMS, 2014. The WoRMS Taxon matcher. Http://www.cmar.csiro.au/datacentre/irmng/.

Spelling variations
Asthenognathas inaefaipes
Asthenognathus inaeqipes
Asthenognathus maefaipes
Astheognathus inaequipes
Asthenognathus inaeguipes
Astheognathus inaeqinipes
Asthenognathus inaequipes

Table 1: Variations on a theme: spelling variations for *Asthenognathus inaequipes*, a crab species from the Varunidae family. All spelling variations were taken from data contributions to OBIS (Berghe et al., 2010). Here, only variations in the name proper are shown; the number of different spellings of the taxonomic authority is often much higher.

Scientific Name as in TAF	Authority as in TAF	Matching Score	ID
Gadus morhua	Linnaeus, 1758	0.91	FISHBASE:69
Gadus macrocephalus	Tilesius, 1810	0.66	FISHBASE:308
Galeus murinus	(Collett, 1904)	0.62	FISHBASE:808
Gadella maraldi	(Risso, 1810)	0.60	FISHBASE:2011
Gadus ogac	Richardson, 1836	0.60	FISHBASE:309
Garra mirofrontis	Chu & Cui, 1987	0.58	FISHBASE:60455
Garra mamshuqa	Krupp, 1983	0.57	FISHBASE:27887
Garra mullya	(Sykes, 1839)	0.56	FISHBASE:24477
Lates mariae	Steindachner, 1909	0.56	FISHBASE:9898
Gadella macrura	Sazonov & Shcherbachev, 2000	0.55	FISHBASE:60872

Table 2: Output of the default BiOnym matching chain on the input string *Gadus morrhua* (Linnaeus, 1758) where the specific epithet morrhua with double "r" is a misspelling and the presences of brackets around the authority is a mistake.

a. Comparison of parsing times

GNI parsing time (ms)	64582
REGEXP parsing time (ms)	219
Number of Inputs	1023
Identically parsed	924
Differently parsed	99

b. Evaluation of parsing results by human expert

	GNI	**REGEXP**
Correct results	5	53
Wrong results	55	9
Uncertain results	39	37

c. Agreement between the parsers

Number of Inputs	1023
Identically parsed	947
Differently parsed	76

d. Evaluation of the parsing disagreement entries by human expert

	GNI	**REGEXP**
Correct results	2	71
Wrong results	73	3
Uncertain results	1	2

Table 3: a. Parsing times and scientific names parsing equivalence for GNI and REGEXP. b. Evaluation by human expert of scientific names parsing by GNI and REGEXP, on 99 disagreement entries. c. Summary of the agreement between GNI and REGEXP on authorities and authority years parsing. d. Evaluation by human expert of authorities and authority years parsing by GNI and REGEXP on 76 disagreement entries.

	Parsing Speed	Parsing Resilience	Accuracy
GNI	**POOR** Being invoked as a remote process, it requires additional overhead for both serialization and deserialization of inputs and outputs, besides the actual time required by the parsing	**GOOD** The parser is quite sensitive to noise and capitalization issues and thus might produce sub-optimal results in circumstances where author names (for instance) are provided in lowercase	**EXCELLENT** Potentially it can return all the taxonomic atoms from kingdom to infra-specific epithets, including authority references and co-autorship.
REGEXP	**EXCELLENT** Being executed locally, on the same machine hosting the parser wrapper, it doesn't require any serialization and deserialization overhead. Also, the parsing itself is fast by design at the expense of accuracy.	**GOOD** Can identify taxonomic atoms out of most of the more common pattern. It is robust to noise in the input data.	**GOOD** As it is designed mostly to be fast and resilient, it focuses on identifying only genus, species and authority information. Other taxonomic atoms are either discarded or misinterpreted.

Table 4: Comparison between the behaviour of two species scientific names parsers. We associated a quality evaluation to three aspects we find crucial for such systems.

a. Performance on Real Misspellings in Percentage				
10 outp. names and REGEXP Parser				
	GSAy	**Taxam.**	**Levensht.**	**Trigram**
Rec.Perc	10.82	64.43	86.02	88.55
Precision	41.90	8.09	8.91	10.05
Recall	11.65	85.93	97.70	97.40
F measure	18.20	14.79	16.32	18.22
6 outp. names and REGEXP Parser				
	GSAy	**Taxam.**	**Levensht.**	**Trigram**
Rec.Perc	10.82	61.82	85.66	88.28
Precision	42.61	11.84	14.66	15.83
Recall	11.65	85.07	97.69	97.39
F measure	18.27	20.77	25.49	27.22
2 outp. names and REGEXP Parser				
	GSAy	**Taxam.**	**Levensht.**	**Trigram**
Rec.Perc	10.72	54.88	82.60	84.90
Precision	47.85	30.87	42.20	43.73
Recall	11.56	83.50	97.62	97.30
F measure	18.60	45.03	58.91	60.32
1 outp. name and REGEXP Parser				
	GSAy	**Taxam.**	**Levensht.**	**Trigram**
Rec.Perc	10.00	43.87	71.39	73.89
Precision	55.80	49.24	72.84	75.68
Recall	10.86	80.31	97.27	96.93
F measure	18.18	60.92	83.28	84.96

b. Performance on Simulated Misspellings in Percentage				
10 outp. names and REGEXP Parser				
	GSAy	**Taxam.**	**Levensht.**	**Trigram**
Rec.Perc	0.66	69.55	94.82	48.57
Precision	13.28	7.97	13.02	33.20
Recall	0.67	98.31	99.38	53.11
F measure	1.28	14.74	23.03	40.84
6 outp. names and REGEXP Parser				
	GSAy	**Taxam.**	**Levensht.**	**Trigram**
Rec.Perc	0.64	67.56	94.75	48.48
Precision	14.24	11.67	19.19	38.76
Recall	0.65	98.19	99.38	52.99
F measure	1.25	20.86	32.17	44.75
2 outp. names and REGEXP Parser				
	GSAy	**Taxam.**	**Levensht.**	**Trigram**
Rec.Perc	0.59	60.88	93.26	47.81
Precision	18.46	30.92	48.95	59.21
Recall	0.59	97.98	99.36	51.98
F measure	1.15	47.00	65.59	55.34
1 outp. name and REGEXP Parser				
	GSAy	**Taxam.**	**Levensht.**	**Trigram**
Rec.Perc	0.41	53.13	90.41	45.02
Precision	21.49	53.84	90.96	83.40
Recall	0.42	97.70	99.34	49.46
F measure	0.82	69.41	94.96	62.07

Table 5: Performance of simple workflows made up of a REGEXP parser and one matcher. Values are calculated on real (a) and simulated (b) misspellings, by varying the maximum length of the output list. The abbreviation "Rec. Perc." indicates the percentage of correctly recognized names. Precision, recall and F measure are commonly used evaluators of Information Retrieval systems. The "outp." number indicates the maximum number of allowed names in the output list.

a. Performance on Real Misspellings in perc.					b. Performance on Simulated Misspellings in perc.				
	Rec.Perc.	**Precision**	**Recall**	**Fmeas.**		**Rec.Perc.**	**Precision**	**Recall**	**Fmeas.**
TaxaMatch	73.18	78.70	89.73	83.85	**Taxamatch**	19.55	83.63	19.93	32.16
BiOnym (10 outp., GNI)	74.96	7.71	97.36	14.28	**BiOnym (10 outp., GNI)**	83.22	11.22	96.39	20.10
BiOnym (6 outp., GNI)	71.25	12.16	97.21	21.61	**BiOnym (6 outp., GNI)**	80.35	16.39	96.17	28.01
BiOnym (4 outp., GNI)	68.01	17.38	97.08	29.47	**BiOnym (4 outp., GNI)**	78.67	22.76	96.02	36.79
BiOnym (2 outp., GNI)	61.68	31.50	96.83	47.52	**BiOnym (2 outp., GNI)**	74.30	40.01	95.70	56.43
BiOnym (1 outp., GNI)	49.22	50.24	96.09	65.95	**BiOnym (1 outp., GNI)**	64.86	67.14	95.02	78.68
BiOnym (10 outp., REGEXP)	78.07	7.98	97.62	14.75	**BiOnym (10 outp., REGEXP)**	92.09	11.00	99.75	19.82
BiOnym (6 outp., REGEXP)	74.59	12.69	97.50	22.46	**BiOnym (6 outp., REGEXP)**	90.25	16.76	99.74	28.69
BiOnym (4 outp., REGEXP)	71.00	18.11	97.38	30.53	**BiOnym (4 outp., REGEXP)**	88.18	23.64	99.74	38.22
BiOnym (2 outp., REGEXP)	64.49	32.89	97.14	49.13	**BiOnym (2 outp., REGEXP)**	82.97	42.63	99.72	59.73
BiOnym (1 outp., REGEXP)	51.89	52.90	96.50	68.31	**BiOnym (1 outp., REGEXP)**	73.36	73.53	99.68	84.63
REGEXP + Levensht. (10 outp.)	86.02	8.91	97.70	16.32	**REGEXP + Levensht. (10 outp.)**	94.82	13.02	99.38	23.03
REGEXP + Levensht. (6 outp.)	85.66	14.66	97.69	25.49	**REGEXP + Levensht. (6 outp.)**	94.75	19.19	99.38	32.17
REGEXP + Levensht. (4 outp.)	85.06	21.76	97.67	35.59	**REGEXP + Levensht. (4 outp.)**	94.55	26.79	99.37	42.21
REGEXP + Levensht. (2 outp.)	82.60	42.20	97.62	58.91	**REGEXP + Levensht. (2 outp.)**	93.26	48.95	99.36	65.59
REGEXP + Levensht. (1 outp.)	71.39	72.84	97.27	83.28	**REGEXP + Levensht. (1 outp.)**	90.41	90.96	99.34	94.96

Table 6: Comparison among the performance of the Taxamatch algorithm (Rees, 2008b) and BiOnym on real (a) and simulated (b) misspellings. The performance is reported at the variation of the length of the output list of transcriptions and of the parser used by BiOnym. We report also the performance of a workflow using only the Levenshtein distance. The abbreviation "Rec. Perc." indicates the percentage of correctly recognized names. Precision, recall and F measure are commonly used evaluators of Information Retrieval systems. The "outp." number indicates the maximum number of allowed names in the output list.

Perc. of Complementary Recognitions on Real Misspellings using 10 outs and REGEXP parser (rows with respect to columns)					
	GSAy	**Taxam.**	**Levensht.**	**Trigram**	**BiOn.WF**
GSAy		1.70	0.00	0.00	0.00
Taxam.	55.31		1.35	0.21	0.098
Levensht.	75.20	22.93		0.18	10.43
Trigram	77.73	24.34	2.71		10.88
BiOnymWF	67.25	13.73	2.48	0.39	

Perc. of Complementary Recognitions on Simulated Misspellings using 10 outs and REGEXP parser (rows with respect to columns)					
	GSAy	**Taxam.**	**Levensht.**	**Trigram**	**BiOn.WF**
GSAy		0.02	0.00	0.14	0.00
Taxam.	68.91		0.27	30.53	0.18
Levensht.	94.16	25.55		46.29	2.99
Trigram	48.05	0.04	9.55		1.46
BiOnymWF	91.42	22.71	0.25	44.98	

Table 7: Percentages of complementary errors committed by one workflow with respect to another. The table reports the percentage of species names recognized by the workflow on the left side with respect to the workflow on the top side.

```
match(I, R) := function(I, R) {
   for each i in I {
    for each r in R {
      score(i,r) := matcher.match(i, r);
       if(score(i,r) >= matcher.RECOGNITION_THRESHOLD)
         results(i) := results(i) U { r, score(i,r) };
    }
         }

    for each result in results {
     if(result.size() > matcher.MAX_TRANSCRIPTIONS_PER_INPUT) {
      result := result.sortByScoreDescending();
      result := result.subset(0,matcher.MAX_TRANSCRIPTIONS_PER_INPUT - 1);
     }
                }
   return results;
 }
```

Figure 1: General algorithm of one matching step, in pseudocode. The differences among the BiOnym matchers are in the way the similarity score is calculated.

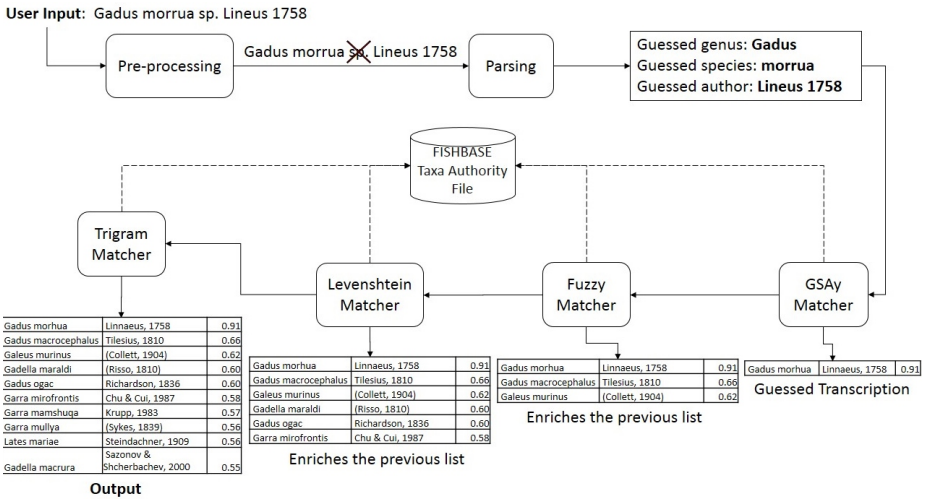

Figure 2: Representation of the work of a BiOnym Workflow, based on the FishBase TAF.

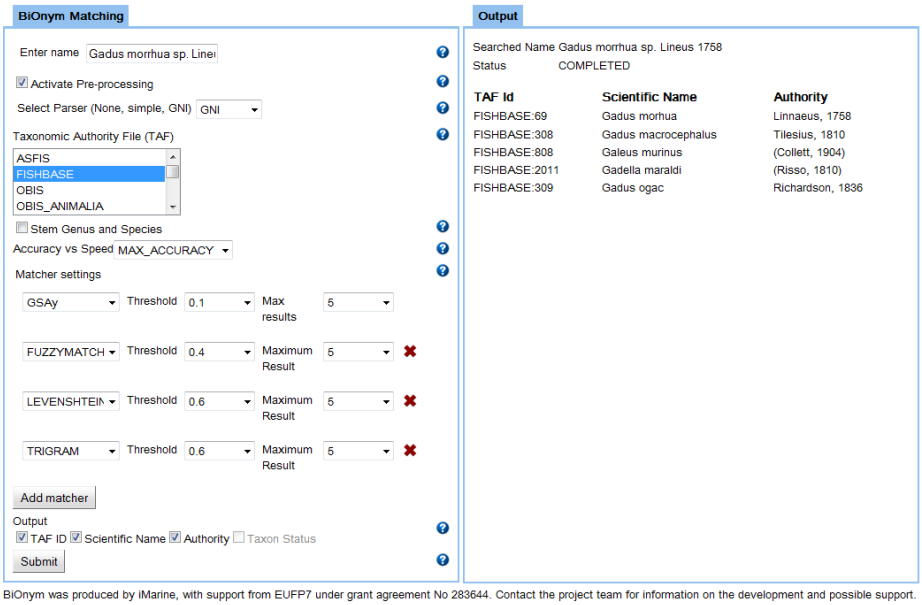

Figure 3: The BiOnym workflow web interface allows users to configure and run the matching chain on the complete scientific name of a species.