NTNU
Innovation and Creativity

Identifying Duplicates
Disambiguating Bibsys

Kristian Myrhaug

Master of Science in Informatics
Submission date: February 2007
Supervisor: Trond Aalberg, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Identifying Duplicates

-Disambiguating Bibsys-

Kristian Myrhaug

28. February 2007

Preface

This thesis is part of the my (Kristian Myrhaug) Master of Science in Informatics degree from the Norwegian University of Science and Technology (NTNU), Faculty of Information Technology, Mathematics and Electrical Engineering, Department of Computer and Information Science, Sem Sælands vei 7-9, NO-7491 Trondheim, NORWAY.

Initially the task which me and my supervisor, Trond Aalberg, gave to me, seemed to be a simple task. But as my knowledge of the poor condition of the content of catalogues in BIBSYS grew, the task suddenly became more complex. (By poor condition a mean that the information of each field in a record is plain text, rather than structured text.) Luckily I was already familiar with the field of *Information Retrieval*, which probably gave me an advantage and also increased my familiarity with it.

Research on the handling of large data, gave me insight into useful methods of filtering and delimiting sets to decrease the number of compares and thereby increase the speed of a search facility. The experience I got through understanding other peoples work on this subject, and trying it out myself, is something that I would be needing at a later stage of my career too.

At the end of the thesis I see that several *Information Retrieval* methods and algorithms could be utilized, and several set theoretic methods could be combined to increase the usefulness of the developed system iDup.

I would like to thank those that have given me support during this process. Especially; Trond Allberg, for beeing my supervisor, supporting me, and providing useful information. Janne L. M. Resby, (my sister) for giving me feedback on written language. Hans I. Myrhaug, (my brother) for fruit full discussions around the various subjects. And Ayse Göker, for *Information Retrieval* papers.

Kristian Myrhaug *Trondheim, 28. February 2007*

Abstract

The digital information age has brought with it the information seekers. These seekers, which are ordinary people, are one step ahead of many libraries, and require all information to be retrievable by posting a query and/or by browsing through information related to their information needs. Disambiguating (identifying and managing ambiguous entries) creators of publications, makes it browsing in information related to a specified creator feasible. This thesis pose a framework, named iDup, for disambiguation of bibliographic information, and evaluates the original edit-distance and a specially designed time-frame measure for comparing entries in a collection of BIBSYS-MARC records. The strength of the time-frame measure and edit-distance are both shown, as is the weakness of the edit-distance.

Table of Figures

Table of Contents

1 Introduction

The number of publications through out the world is increasing rapidly, with it follows the information seekers' increased need to link relevant information together. Two publications can be related to each other in several different ways. Some of the ways to compare publication are based on document content itself, while others are based on external annotation of the document (e.g. metadata).

The content-based information retrieval methods are often used on documents where there is a lack of annotations present, while the content annotation methods are often used on documents which have a good portion of metadata available. Both approaches can also be combined into a hybrid method, in which information within the document and external annotations are used to help information seekers find relevant publications.

In general, people seek information when there is an abnormality between their own knowledge and the knowledge needed to help solve their problem at hand or simply to satisfy some interests and desire for information. The information source can be everything from a person trying to shake his/her head a bit to remember some forgotten knowledge, other people, books, internet, libraries, or other interactive media.

Different organizations and public libraries gives information services to their users. BIBSYS, is such an organization, that serves as a common library repository for the educational institutions of Norway. Each institution has to take care of their own printed and digital material, but their library catalogue are public to all the other institutions in the organization. This is a great way of distributing information and allowing loan between the institutions. And brings a larger information base to the individual seeker.

The work in this thesis is motivated by the seeker trying to find information in a library context, in which catalogues and categorization has been the most commonly used method to help information seekers find the right information. Libraries have books that are currently mostly printed. This implies that the catalogues are annotations of printed documents, although this century would most likely bring a gradual shift from printed publications to more digital publications too. Some libraries are much ahead of other libraries in their digitalization process.

The problem this thesis aims to solve is that the information seekers are sometimes loaded with the unnecessary burden of verifying ambiguous information items - e.g. duplicates. Given the digital library vision above, the thesis aim to provide a framework for disambiguating catalogues for digital libraries. Examples of record fields in a typical library catalogue can be: individuals, corporation and convention fields in bibliographic catalogues. Typically, there are several fields in a record, and ambiguity can be found within and across/between each of the field type in the catalogue.

2 Thesis Overview

2.1 Problem

The problem in this thesis is to automatically or semi-automatically disambiguate creators[1] of publications in the bibliographic records of BIBSYS. This indicates building a system where one could easily replace ambiguity measures[2], and evaluate compositions of different measures that form a compound ambiguity measure.

2.2 Context

The source of bibliographic records are a set of MarcXchange files harvested from BIBSYS-MARC records found in BIBSYS. These BIBSYS records distinguish the entities responsible for publications (creators) into three categories; *individuals*, *conventions* and *corporations*. Further notion of the BIBSYS-MARC format will be provided in a later section (section 3.3.1.1).

2.3 Overview

The remaining of this thesis is organized in a structure where the reader is first presented with *Background* information that is necessary or important to fully understand concepts described at later stage in the process.

The *Background* section gives a foundation to read the *State of the Art* on disambiguation, which is the section immediately following the *Background*. *State of the Art* gives an objective insight into the subjects that are directly or closely related to this thesis.

The *Background* and the *State of the Art* are objective descriptions of research on the different subjects. These descriptions are discussed and given some thoughts in the *Bibliographic Discussion* section. The Bibliographic Discussion section also gives some input on iDup's[3] perception of the information it is about to explore.

To keep the reader on track, the *Methodology* section describes how the problem is approached to build a modular system, that is capable of disambiguating records with some replaceable measures.

The *Methodology* section is followed by *Content of BIBSYS. Content of BIBSYS* describes observations on the content of relevant fields in the records of BIBSYS. These observations should be helpful when selecting matching measures.

Matching measures and other modules that form the whole system are described in the *Solution*

1 Creator is an entity responsible for the creation of a document/object.

2 Ambiguity measure: A measure that decides whether two or more entities are ambiguous or not.

3 iDup is the project's name.

section. Some of the module implementations are direct results of the previous *Content of* BIBSYS section. The modules are described in detail, and the overall information flow within the system is also described.

The design of the system, and the chosen measures has to be evaluated in some sense. How it is evaluated, and the evaluation itself are described under the *Evaluation and Discussion* section that follows the *Solution* section. It will highlight positive and negative experiences, and recommend small changes that would improve the totality of the system. The evaluation brings us to a *Conclusion*.

3 Background

3.1 Information

A widely adopted interpretation of what information is, has evolved from the poem *The Rock*, by T.S Eliot [Eliot, 1934]:

```
Where is the Life we have lost in living?
Where is the wisdom we have lost in knowledge?
Where is the knowledge we have lost in information?
```

This poem was later recognized, by Harland Cleveland [Cleveland, 1982], as a hierarchical structure of information, knowledge and wisdom. Later this hierarchy has expanded to include data, and was called the Data-Information-Knowledge-Wisdom (DIKW) hierarchy, where data is the building blocks of information, information supports knowledge, and understanding of knowledge gives us wisdom. In this thesis the content of fields are of interest. Content of fields would be the data and information parts of DIKW:

- **Data** is the terms and notation in a field, and
- **Information** is the meaning of all data combined in one field.

Examples of data items are:

```
Myrhaug
,
Kristian
Asle
```

But when they are put together and it is given that this is a name, then this becomes a piece of information. This information can be enhanced with for example the knowledge that anything before a first comma is/are surname(s).

```
Kristian Asle Myrhaug
```

A large piece of information would then be a document or another kind of object, dependent on the building blocks it consists of.

3.2 Document like Objects

Documents are objects in the physical or digital world, that can be described in a digital catalogue. Or as Buckland [Buckland, 1997] described it: "Digital systems have been concerned primarily with text and text-like records (e.g. names, numbers, and alphanumeric codes), but the present interest in icons and graphics reminds us that we may need to deal with any phenomena that someone may

wish to observe: events, processes, images, and objects as well as texts". Seekers' interest in these objects brings us to managing them.

3.3 Digital Libraries

The current conception of Digital Libraries are strongly related to the technology, the daily management and the users' tasks. The current technology is for keeping track of the documents, and to display those that are digitalized. The daily management is serving the users' needs for information and preserving the documents it contains, and the work performed by users are expected to occur individually.[Levy-Marshall, 1995]

Until recently, development on digital libraries has been attempting to mimic specific functions of conventional libraries. This aim to support, or automate, the work (work as described by Levy and Marshall) involved in a library, has reached a level where it is appropriate to examine the roles a library could play in the community. And from this examination develop additional functionality to support the roles. [Besser, 2002]

One of these roles is supporting the information seeker to retrieve all relevant information to his/her query. Yee [Yee, 2005]explains the commonly inability to retrieve all information on a work, due to differing versions of citations. The differing versions of citations can have its background in import from another library where the authority files (section 4.1.1) can be differing, have no common identification scheme, or the authority control was not present when the first records where added to the library catalogue. The internal storage and/or the communication between two or more libraries are often in the well known MARC format.

3.3.1 MARC format

MARC is a short for MAchine-Readable Cataloguing. MARC is a record structure designed for bibliographic information. This record structure often follows a ISO-standard (ISO 2709) coding to separate the different fields from one and other. The MARC format is an exchange and registration format for bibliographic records. The MARC record structure is widely used in both national and international libraries across the world, and support a detailed description of document like objects. It was originally developed by the Library of Congress in the 1965 - 1966 time period, but through its generations got several distinct dialects. One of these is BIBSYS-MARC. [FRBR-BIBSYS, 2005]

The MARC format consists of control fields and variable fields. Control fields are of fixed length, and have a controlled name space where the importance of correct character positions are a matter of necessity. When one is talking about MARC fields one are referring to *data fields*[4] annotated by a three character code consisting of digits, this gives a range from 000 - 999. These *data fields* are also divided into *sub-fields* with the range of, $a - $z, and $1 - $9. [FRBR-BIBSYS, 2005] The *data*

4 Here the data fields is a misleading terminology, in contrast with the DIKW hierarchy.

fields has a logical structure, where each is referring to an entity that has a role/relation to the work. This role is not described in the records itself, but in written manuals for registering records. The fact that these manuals change from time to time is probably a source of incorrect registered entities. Sub-fields describes attributes of entities in the record. The content of such an attributes, can consists of one or several *data items*[5].

3.3.1.1 BIBSYS-MARC

BIBSYS-MARC is a dialect of MARC, that evolved in the academical community of Norway. As mentioned earlier the original MARC format followed the ISO-2709 standard for coding records, but BIBSYS-MARC never followed it. BIBSYS-MARC is not only used for exchange between the different members of the BIBSYS, it is also used for storage. BIBSYS-MARC is the main storage format of the BIBSYS system. [FRBR-BIBSYS, 2005]

In the BIBSYS-MARC records, individuals can be found in 100, 600 and 700, corporations in 110, 610 and 710, conventions in 111, 611 and 711, and finally publications in the 240, 241 and 245 *data fields*. Actually they are found in other data fields too, but these are the ones that are most used. *Names*/*labels* are found in the *sub-field* $a, *time-frames* in $d, and *identifiers* in $5. [BIBSYS-MARC, 2006] Here is an example:

```
*001941252426
*008  $ap
$bv
$cnob
*080uk$a839.6
*082uk$a839.82
*100  $aIbsen, Henrik
*240  $aVerker
*245  $aSamlede verker
*250  $a9. Utg.
*260  $aOslo
$bGyldendal
$c1941
*300  $a5 b.
*691**$askjønnlitteratur
```

3.3.1.2 MarcXchange

The record that where harvested from BIBSYS in 2005, consisted of a simplified MarcXchange format. The simplified MarcXchange [MarcXchange, 2006] format has the root tag `<collection>` which can contain several `<record>` items. The `<record>` item can contain a `<leader>`, several `<controlfield>` and several `<datafield>`. The *leader* and the *control fields* are not of interest in this thesis, but the *data fields are.*

5 Here data should be interpreted as in the DIKW hierarchy.

A *data field* correspond to one entity. Such an entity has a predefined role that the BIBSYS-MARC defines by a three digit code , this code is an attribute named *tag* in the XML scheme. A data field could also contain several `<subfield>` elements, that has a *code* attribute corresponding to a *sub-field* in the BIBSYS-MARC format. A *sub-field* contain different types of information about the *data field* entity that contains it. MarcXchange in general (Illustration 1):

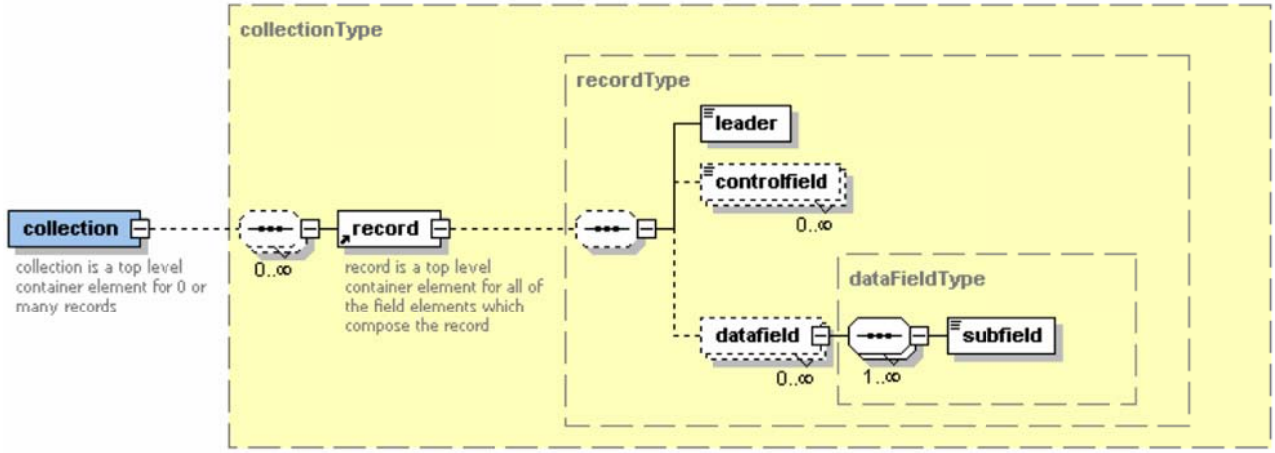

Illustration 1: MarcXchange XML scheme [MarcXchange, 2006]

3.4 Information Retrieval

Information retrieval (IR) is concerned with finding and retrieving information relevant to a given query. There are several well known IR models, like for example the *Boolean*, *Region*, *Vector*, *Probabilistic* and *Levenshtein distance* models. Common to all the IR models is that they are meant to find similar information. In bibliographic context this information is often extracted from either a unstructured, semi-structured or structured source.

3.4.1 Text

As mentioned earlier, text can be either unstructured, semi-structured ore structured. The simplest is unstructured text.

3.4.1.1 Unstructured Text

Unstructured text, or more often called plain text, is just text. “In its most basic form, text consists of plain alphanumeric characters. The most common representation of it is the American Standard Code for Information Interchange” [Lu, 1999]Plain text does not give away any semantics related to the context nor the content of the document. To gain the advantage of semantics, one has to analyse

the text thoroughly.

3.4.1.2 Semi-structured Text

Adding some styles and attributes to a text, gives us a semi-structured text. An example of semi-structured text is Hyper Text Markup Language (HTML). In HTML we find a limited range of annotations concerning document properties, and limited structure of style formats on the content of the document.

3.4.1.3 Structured Text

Structured text consists of a more complete annotation scheme, that often better support more advanced displaying than plain text or semi-structured text. Examples of structured text formats is, XML, XHTML and SGML. Structured text formats can often be nested into substructures, where one can express, for example; That a catalogue is part of a library, and this catalogue has a label and an identifier, and that the catalogue is filled with catalogue cards, with the label... and so on.

3.4.2 IR Models

The IR models that accept retrieval of information which are not exact matches are those that are interesting for this thesis, because names and words could be misspelled or have abbreviations that can not be detected by a IR model that has a true-false comparing result.

3.4.2.1 Boolean Model

The boolean model is one of the models that has a true-false outcome. Its queries are annotated by the operators AND, OR and NOT. These operators and brackets, make up a powerful language to express one's needs. But “...most users find it difficult and awkward to express their query request in terms of Boolean expression.” [ModernIR, 1999]

3.4.2.2 Region Model

The region model is an extension to the boolean model, it makes it possible to retrieve information that has a region that satisfy a region described in a query. The query, in addition to the boolean operators, has at least the two operators CONTAINED and CONTAINED_BY. [Hiemstra, 2007] This gives us the opportunity to ask for a specified title by stating a query like this;

```
<title> CONTAINING 'The specified title'
```

A HTML document that satisfy the above query could look like the following.

```
<html>
  <head>
```

```
    <title>The specified title</title>
  </head>
  <body>
    <p>The first paragraph</p>
  </body>
</html>
```

3.4.2.3 Vector Model

In this model each word has its own axis in a n-dimensional space. Therefore, every document/query would be represented as a vector that is almost unique[6]. When two vector are compared, one can calculate a variety of relationships between the two. The most widespread relationship to use as a similarity measure is the cosine angle [Salton, 1971] of the two vectors, which has an outcome between 0 and 1. The distance of two vectors are often used to find clusters within a document collection. The well known Inverse Document Frequency (IDF) is widely used to give important words more score than less important terms.

3.4.2.4 Probabilistic Model

The probabilistic model utilize probability theories to estimate the best ranked set response to a user query, by using all information available when the query was posted. [Robertson, 1977] Hiemstra [Hiemstra, 2007] explains what impact Robertson's contribution has; Suppose a single query term `social`, and a document collection of 10000 documents, where 1000 documents contain the `social` term, 9 documents are relevant without containing the `social` word, and 1 is relevant and containing the `social` term (Illustration 2). The probability (0.0010) of selecting a relevant document from the set containing the `social` word, is less than the probability (0.0011) of selecting from the relevant document in the opposite set.

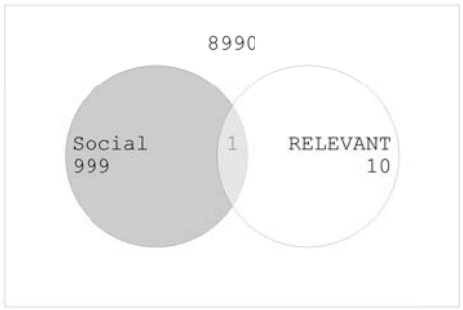

Illustration 2: Venn Diagram [Hiemstra, 2007]

6 The sequence of terms are not detected, unless the model is extended.

3.4.2.5 Edit Distance

The Levenshtein distance is generally known as the edit distance, and is part of a breed of algorithms that is calles approximate string matching. “The edit distance between two strings, is the minimum number of character insertions, deletions, and replacements needed to make them equal.” [ModernIR, 1999] Thus, if one considers a document to be a string of characters, then this would be one measure for the distance between two documents.

4 State of the Art

4.1 Controlling Ambiguity

In contrast to management systems that do not allow several different entries of a specific entity, there are systems that allow multiple entries for one entity. A system that allows multiple entries describing the same entity, should also keep track of the entities that has duplicate/ambiguous entries. In bibliographic databases there are at least two ways of controlling ambiguity.

4.1.1 Authority Control

As Harter describes it, “Authority control is a key feature, in which names of authors, variants of works (editions), and subject headings or descriptors are all controlled. The concept of authorship and ownership are extremely important in a traditional research library, in which various forms of an author's name are brought together in a name authority file.” [Harter, 1997]

4.1.2 Access Control

A model on handling ambiguous/abbreviating person names in libraries has been proposed by others and tested for feasibility by Snyman and Rensburg. [Snyman-Rensburg, 2000] To understand access control it is useful to see it in contrast with authority control. When authority control is used for controlling ambiguity, one form of the name is chosen as the authoritative form. In contrast to the authority model, the access control model links the variant forms without any form being selected as the authoritative. The access control model ensures equal importance of every form of the name. The access control model can be achieved by giving each person a unique identifier, called a INSAN, which can be linked to a variety of other proprietary identifiers. Snyman and Rensburg concluded that the access control model would revolutionize authority control in libraries.

4.2 Disambiguating

As we saw in the Digital Libraries section, the authority control mechanisms are failing or not utilized properly. Therefore there is an increasing need to identify ambiguous entries to relief the seeker from the task of identifying the aquired entity.

Previous work on the disambiguation of bibliographic information are closely related to *record linkage*, *the merge/purge problem*, *duplicate detection*, *hardening soft databases*, *reference matching*, and *entity-name clustering and matching*. [Bilenko-Mooney, 2003] Most of these are not restricted to the bibliographic domain, but can be restricted to specified information types[7]. Projects that arose in the library domain are; *VIAF* (Virtual International Authority File), *LEAF* (Linking and

7 Information types; surname, street address, phone number, etc.

Exploring Authority Files) and the *InterParty Project*.

4.2.1 Record Linkage

In *record linkage* [Winkler, 1999] one utilizes a score which is called ratio. The ratio, or odds, between two record fields are evaluated by calculating the probability of them being a link/match, given some knowledge of the total population for the specified domain/information type. Such a knowledge of the total population could for example be the total occurrence of a specific surname.

When selecting which two records are a match, one splits the result into *matches*, *possible matches*, or *non-matches* by setting two thresholds. The threshold and possible matches could be evaluated automatically, semi-automatically or manually.

4.2.2 The Merge/Purge Problem

The Merge/Purge Problem is concerned with merging large databases. Hernández and Stolfo [Hernández-Stolfo, 1995]saw a time consuming process, and where looking for a balance between maximum true record matches and minimum time line. They presented two ways of doing this:

The first was by creating keys/fingerprints, then sorting the data records on those keys, and later slide a *window* sequentially over the records while matching within these records. This made it possible to keep a *window* in main memory while matching within it. Narrowing the *window* size would result in less true matches, and widening the size result in more time cost.

The second method also started out with creating keys reflecting the content of the record. This *n* size key where then mapped into a *n* dimensional space where a clustering algorithm was applied. The theory is that every cluster would become small enough to keep in main memory, and then merge records within every cluster.

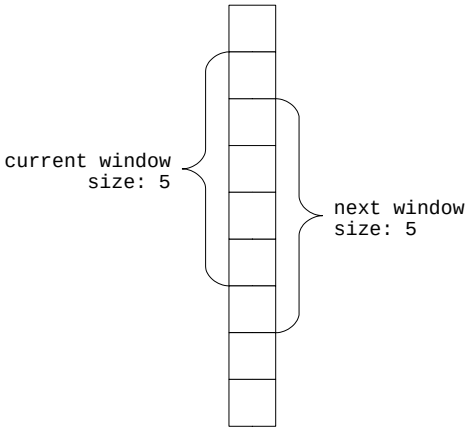

Illustration 3: Sliding window

4.2.3 Duplicate Detection

Monge and Elkan [Monge-Elkan, 1997] tells us that the standard way of detecting exact duplicate records for the same entity is to sort the database table, and sequentially traverse it while checking for identical rows. They also point out other ways of grouping similar records, for example by querying the collection. The sequential traverse of sorted records method could be extended to allow approximate duplication detection. An example of this is the work of Hernández and Stolfo [Hernández-Stolfo, 1995], as we saw in the previous section. In Monge and Elkans duplicate detection, they check for duplicates in several passes and stretches the window, which they named *priority queue*, more dynamically. The latest matched entities are kept in main memory in a queue[8] with a fixed size. When matching they used a variant of the Smith-Waterman [Smith-Waterman, 1981] edit distance.

4.2.4 Hardening Soft Databases

A soft database is a database where distinct records can reference the same real-world entity, in contrast to a hard database where only one reference to a real-world entity is allowed. Cohen, Kautz and McAllester explains that hardening a soft database, is to build the unknown underlying hard database by finding co-references in the soft database. [Cohen-Kautz-McAllester, 2000]

In the example below we see that a source record, *1*, has a reference which match with a reference

8 A better terminology is a FIFO cache.

in the second record, *2*. This brings us to the conclusion that a real world entity “Kristian Asle Myrhaug” has three different works; “Identifying Duplicates – Disambiguating BIBSYS”, “Java Source Code” and “Compiled Java Bit Code”.

```
ref(creator, record number, work)

ref(“Kristian Asle Myrhaug”, 1,“Identifying Duplicates”)
ref(“Kristian Asle Myrhaug”, 1, “Java Source Code”)

ref(“Kristian Myrhaug”, 2, “Identifying Duplicates – Disambiguating BIBSYS”)
ref(“Kristian Myrhaug”, 2, “Compiled Java Bit Code”)
```

4.2.5 Reference Matching

Reference Matching is bound to match citations in the reference section of papers. McCallum, Nigam and Ungar experimented by first splitting the data into overlapping sets, *canopies*[9], by using a efficient clustering/splitting algorithm (the *vector space* model). After the first clustering step, they could apply any clustering algorithm to the individual *canopies*. This restricts distance measuring to be contained within the individual *canopies*, which imply less distance measuring on the total data set. [McCallum-Nigam-Ungar, 2000]

4.2.6 Entity-name Clustering and Matching

Entity-name matching is defined to be “the task of taking two lists of entity names from two different sources and determining which pairs of names are co-referent (i.e., refer to the same real-world entity).” [Cohen-Richman, 2002] Entity-name clustering is defined “as the task of taking a single list of entity names and assigning entity names to clusters such that all names in a cluster are co-referent.” They achieve adaptive matching by learning[10] how to match a cluster.

4.2.7 VIAF

VIAF is a project where Die Deutche Bibliothek, the Library of Congress and OCLC Online Computer Library Center are cooperating “to prove the viability of automatically linking authority records from different national authority files and to demonstrate its benefits.” [VIAF, 2006]

The VIAF solution is done by first creating Enhanced Authority records from both libraries, and then matching these records to find Enhanced Authority records which are about the same person. Enhanced Authority record is a record of all relevant information, about one author, found in one library catalogue. The matching is done by a score, where different criteria are weighted into strong, moderate or week support.

9 A canopy is a cluster in an n-dimensional space, that could overlap any other canopy.

10 Machine learning.

4.2.7.1 Comparing names

When matching person names in VIAF, the question is simply: Is the names compatible? In more technical terms, this means that a persons name is split into sub names (first name, middle name, last name), and compare with the other record separately. If one of the sub names are not compatible, the whole name is not compatible. Two sub names are compatible if the longest of them starts with the shortest of them.

4.2.7.2 Confirming a match

In VIAF, when a person name compatible with another person name is not sufficiently proof that it is the same person, and supplemental information is needed to confirm the match. Similar titles, correlation with dates, and co-authoring are some of the supplemental information which could strengthen/confirm a match.

4.2.8 LEAF

Linking and Exploring Authority Files is a project which link authority records together, based on a query. A user query is posted and the members of MALVINE are queried. The result of those queries are stored as a entry in the common MALVINE authority file. A good feature of this linking is that a “user may offer additions, corrections etc. to LEAF; the data will be checked by an Intellectual control agency”. [Weber, 2002]

4.2.9 InterParty Project

The InterParty Project [InterParty, 2003] seeks to provide a service for parties in a network. The service should be able to exchange bibliographical and rights management information on different publications. To do this the service need to disambiguate the different records of the parties. The disambiguation deals with public identities, like for example a real world person with two different aliases are perceived as two separate entities, unless there is a third entity linking them together.

4.3 Future of Libraries and Digital Libraries

4.3.1 Functional Requirements for Bibliographic Records

FRBR is a recommendation of IFLA (International Federation of Library Associations and Institutions) to restructure catalogue databases into a concept of entity-relationships between publications, creators, and subjects. The theory is that the recognized entities can be sorted into a model where "a work is realized through one or more expressions, each of which is embodied in one or more manifestations, each of which is exemplified by one or more items". This model would make it easier to retrieve relevant information through one of the four entity relations. [FRBR-OCLC, 2006]

4.3.2 DOBIS-LIBIS

"The set of all entries of a particular index type in a catalogue is called an authority file because this list contains the authorized version of the entries." [McAllister, 1981] One characteristic of a bibliographic database is the necessity for maintaining uniformity of keys (authority files). The reason for this necessity is a retrieval issue, where names, titles etc. can be misspelled or have different forms which conceal this entry from users. [McAllister, 1981]

4.3.2.1 Document and access-point files in DOBIS-LIBIS

To support the requirement, the keys of the records are not stored in the records themselves, but in separate files which becomes the index files. The record themselves get a unique access-point (identifier) assigned to it for direct access, while the index files holds these access-points. This method of authority control is somewhat similar to Access Control, and also very relevant to information retrieval, because of the index which are a lookup (e.g. searchable).

4.3.3 MAVIS 2

The second version of Multimedia Architecture for Video, Image and Sound is an "..open architecture that supports content based multimedia information exploration." [MAVIS2, 1999] This architecture supports the concept of feature vectors (called signature) to store and retrieve managed documents/objects[11]. This signature is be fairly unique for each object, and also reflects the object type (image, video, text etc.). A “synonym” mapping between the different object types spaces, gives the system a capability to get request in one object type, and respond with any other object. For example, a user could query the textual word “red”, and get pictures that has a mean histogram value of red. The features are extracted from media type specific processors (off-line).

When concerned with conventions, corporations, individuals or publications, they could be considered to be in separate media space with a own signature space, for example INSAN and some other features (for ease of retrieval).

11 For text, this would be a feature vector from the *Vector Model*

5 Bibliographic Discussion

The *Background* and *State of The Art* sections above gave information that is necessary to understand the design decisions made during the development stages. But to give an even better explanation to the design, this section will link the bibliography with the early development stages.

The Background section described the context of the thesis, where BIBSYS-MARC and MarcXchange was key subjects. The communication and storage format of BIBSYS has its on practice of registering records in its data-fields and sub-fields. These were of course reflected into the MarcXchange XML files which were the record source for the system.

Looking at the structured text in the MarcXchange files, we find fields of various types (leader, control-field, data-field and sub-field). The fields of interest are already chosen to be data-fields and sub-field with specific attributes. But what is really a field?

5.1 A Field

Perhaps the most applicable way of thinking about a information retrieval is probably by thinking fields, where both annotation items and content items are treated in the same way. Or in terms of a database: *author* is a column, *abstract* is a column, *content* is a column, and so on. From a markup language point of view, this means that the content of a tag, should be treated like information that is just as important as any other tag content. This is shown below.

```
<fields>
  <title>A Title</title>
  <individual>An author</individual>
  <abstract>The abstract</abstract>
  <content>The content (an image or text)</content>
</fields>
```

Most information retrieval (IR) systems only handle a subset of these fields. Like for example in libraries, where a large portion of the content can be non-digital, which imply that the content fields are not easily available fields - and hence either need to be keyed manually or scanned for editing. This is illustrated in markup below.

```
<in-library-record>
  <title>A Title</title>
  <individual>An author</individual>
  <abstract>The abstract</abstract>
</in-library-record>

<not-in-record>
  <content>The content (image, text, etc.)</content>
</not-in-record>
```

You may add to your thoughts that the content of one field, could also be a document like object. For instance an individual name/label can be divided into sub-names, and classified to be first-name, middle-name, surname, etc., this means that the document (the combination of sub-names) can be described by for example a name catalogue (see below).

```
<individual>
  <first-name>Kristian</first-name>
  <middle-name>Asle</middle-name>
  <surname>Myrhaug</surname>
  <initials>K A M</initials>
  <terminals>N E G</terminals>
</individuals>
```

If the content at hand was this well structured, it would have been easier to compare different records to each other, unfortunately its just free-text fields, but we are lucky enough that names are separated from other fields (like for example a publication title). A future system would therefore be able to threat the different information types that the specified fields are suppose to contain.

5.2 Information Structures and Fields

A structured document can be thought of as a collection of fields, where one field can contain many other fields (and some attributes), thus creating a tree of nodes, where every node can have some properties. The MarcXchange files are structured text files, and makes up a better semantic than plain text, but it does not quite relief the burden of analysing some text, even though it is in a much better condition than a plain text source would have been.

The different fields of the MarcXchange source consist of free-text. This type of text are generally difficult to extract useful information from, but by investigating a sample of the population, one can build a fairly good “picture of the world”. This “picture of the world” should be utilized by some means, for example preprocessors. In the most general sense, fields are items that can be part of any kind of text, and have some properties.

5.3 Filtering Entries

In *record linkage*, *the merge/purge problem*, *duplicate detection*, *hardening soft databases*, *reference matching*, and *entity-name clustering and matching* one was trying to solve several problem. One of these problems was how to avoid unnecessary comparing of records.

Monge and Elkan mentioned several ways of filtering unnecessary information. One of these ways, which the VIAF and the InterParty project implements, was by querying the whole collection for records similar to the query. This method is more time conduming than the one they chose themselves, but it ensures that every similar record are being compared, and possible matches are not left out.

5.4 Matching Entries

The disambiguation projects that we have looked at all agree that there has to be some measure that decides when two entries are duplicate, or not. Winkler show a abbreviation of it, where the two entries could also fall in to a *possible match* category. This category could be analysed further by a machine or manually.

When a conjunction of measures are chosen, something or someone has to give parameters to what is a match and what is not. Some approaches to this are by machine learning, while others are by exploring the information type it is set to match. The machine learning method were not appropriate, because it could cause unexpected results if the training set is not a uniformly distributed subset of the population.

5.4.1 Measuring Labels

Label measuring, or name and title measuring, are most often done by a well known IR approaches of either *probabilistic*, *vector space*, or *string approximation*. The probabilistic approach where less represented than the other two. The IR vector space model are more suitable for longer character strings than the approximate string matchers. The position of terms in convention, corporation, individual and publication labels are also very important in the bibliographic domain, and this is where the *vector space*, and the *approximate string* matching approaches differ. *Approximate string* matchers take into account the position of terms, and characters, in the character string, which the *vector space* model does not.

5.4.2 Measuring Dates

Not much are said about using dates associated with records as a possible factor of confirming two entries to be a match, but it is. This thesis will explore the dates, or time-frames, extensively to show the usefulness of them.

6 Methodology

The previous section gave some input on what the disambiguation application should be able to do, and how this should be done. The main topics that arose, was the exploration of samples from the total population, loading records, preprocessing fields, filtering records and matching entries.

There are some dependencies among the above topics. For example a preprocessor must have some knowledge of the information type and structure it should process. To make this knowledge available, one has to explore the content of the records at hand. This will be performed in the next section, *Content of BIBSYS*.

After the *Content of BIBSYS* exploration one is able to implement the different modules as a *Solution*. The *Solution* section gives a thorough description of these modules and the overall communication flow within the system.

A complete system needs to be tested and evaluated. Testing each module was done during the development of it, but *Evaluation* of the system as a complete composition of modules is described after the *Solution* section.

Finally the *Evaluation and Discussion* gives the basis for a *Conclusion*.

7 Content of BIBSYS

Exploring the whole collection of about 4 million records seemed too time consuming, therefore a subset of 100 thousand records was selected from the total records. The sample collection was first loaded into a MySQL database, which one could query and traverse by using any SQL client. Browsing through the database, looking at first letters, showed that all letters was represented. Letters that are expected to contain more entries, did contain more entries. These two facts leads to the belief that the collection is a random selection of the total population.

7.1 Time Frame

Date fields in BIBSYS-MARC, are not composed of any standardised date formats, like for example dd/mm/yyyy. The time-frames are rather free-text that make matching more complex. For this reason it is proper to observe and organize the observations, and later utilize this knowledge.

During the inspection of the date fields, there was three main topics that arose,;the first one was the fields containing not dates at all, the second topic was numbers (e.g. days, months and years), and the finally notations.

7.1.1 Notations

At first glance, one observes that the dominant entries are the once containing either one or two years. Some of the dates containing one year, also has indicators that they are end-dates. There are three notations of that kind ; a leading “-” a leading “d.”, or a “død etter”. Statistics on these, and other notations can be found in the table below (Table 1).

Notation	Comment	Count
a.d, e.kr	Anno Domini (Norwegian: etter Kristus)	9 (2 + 7)
b., f., fl.	Born (Norwegian: født)	43 (6 + 19 + 18)
b.c, f.kr	Before Christ (Norwegian: før Kristus)	23 (15 + 8)
Cent., årh.	Century (Norwegian: århundre)	30 (16 + 14)
ca., “?”	Cirka	69 (32 + 37)
d., leading “-”, død etter	Death	31 (29 + 1 + 1)
or, “/”, eller	Or (Norwegian: eller)	17 (6 + 10 + 1)

Table 1: Notations in time-frame fields

The “d.” notations above, are constrained to time-frame fields containing one year, but there are also other notations which apply to fields containing one or more years. The indicator most related to the “d”, is the “b.”, this notation imply that the following is a start-date. The “b.” notation, might be redundant, because it is assumed that the date is a birth, if nothing else is given. “A.D”, “B.C”,

“cent”, “ca“ and “or” are other notations that are found.

The notations that are found also has some sequence patterns. For instance “A.D” and “B.C” are left directional, this means that the notation applies to everything on its left, until one or the other notation are found. “cent” applies to everything on its left, “ca” can be a bit confusing, because it seem to concern both directions depending on whether it is an ”?” or a “ca”[12]. But the general rule is that “?” applies to the nearest number, and “ca” concern everything on the right of it. “or” applies to the first number on its right.

7.1.2 Days, Months and Years

It is easy to say that all numbers can be either a day, a month or a year, but it is more difficult to separate them from each other when you know that some of them could be in the same numerical range. For example, the numeric value of months is a subset of days, and days is a subset of years.

In most cases the date fields only contain years, this we know because a number larger than 31, could neither be a month, nor a day. So, the real task here is to find patterns within the fields which possibly contains days and months, and within the fields that have notations.

The first pattern is indicated by “b.c”, “a.d” and “cent” notations, which tell you that the information is very old and therefore blurred -hence day or month can certainly not be present in that time-frame. Another pattern is that if a month, in its character range is found, then you will not find a month by a numerical value in that field[13]. A third pattern, postulates that an “or” after a year, indicates that the number on the right of it is a year. This pattern also applies to days and months (dd or dd, mm or mm, and yyyy or yyyy). The final pattern involves inspecting whether the remaining numbers can be days, or months. If we start a counter at 1 on the leftmost number entry, one finds that if the even-count numbers can be months, they correspond to the start-point and end-point month respectively - Otherwise, if uneven-count numbers can be months they correspond to start-month and end-month respectively. If non of the two cases above are found, then you need to check if the last number can be month, if so the other numbers are days. And the last resort is to check if the first number can be a month, if it is the other numbers are days. When non of the patterns above applies, you are probably looking at a sequence of days. Of course if one pattern is present, another is not.

The “cent” notation also has another feature, combined with it. In English one has a suffix on centuries. Example of this are “2nd“, “3rd“, “14th“ and “21st“.

12 Both “?” and “ca” are considered to be a *uncertain* notation. Other notations can also have several instances that has the same meaning, like for example “-yyyy”, “d. yyyy” and “død etter yyyy”.

13 Month's character range is “jan”, “january”, “feb”, etc. month's numerical range is 1 through 12.

7.1.3 Not Dates

When the date field does not contain a date, it is either because it is not known, or because it contains the content of another field. Here are some examples of content from a different field:

```
“23-79 j it.”
“Member, Commission of the European Communities”
“Mrs.”
“London”
“|965”
```

There are 93 occurrences of this, although one of them seems to be a typo (“|965”).

7.2 Label Fields

The label/name fields that where under the magnifying glass was *individuals*, *corporations*, *conventions* and *publications*. As for time-frame fields, the label fields should be normalized/preprocessed to ease a matching process. When talking about normalization it is necessary to say something about the characters that the label should contain. Ideally a label should only contain *letters*, *digits* or *white space* (for ease lets call them normal characters, and the opposite for special characters). In the test collection there where 219765 label fields. Both the labels with normal and special characters are further investigated.

The normal character fields consisted of only the name itself, while some of the special character fields contained information that where not part of the name itself. For example a corporation label could contain a comma notation before a corporation type[14], this indicates that it should have been registered in a different field than the label field (examples below).

```
“Engineering Information, Inc.”
“Maarud, AS”
“Bygg- och transportekonomi, AB”
```

Because the fields with only normal characters are already normalized, they where less interesting than the name fields containing special characters. The different label classes (e.g. individual, convention, corporation and publication) of the special character fields where investigated separate because they contained different type of information.

7.2.1 Individuals

The test collection consisted of 103996 *individual entries*. Table 2 Shows the distribution of special characters. Since the special character set was not normalized, its was proper to look for patterns in these fields.

14 E.g. Inc., Ltd., AS, AB

Character	UTF-8 Value	Occurrence
“	34	3
#	35	7
‘	39	704
(	49	22
)	41	22
*	42	1
,	44	101537
-	45	3856
.	46	25117
/	47	13
:	58	10
;	59	1
<	60	7
>	62	8
?	63	19
[	91	7
]	93	7
‘	96	6

Table 2: Special characters in individual fields

The distribution of special character occurrence, as you can see in Table 2, shows us that there are four special characters that could have significant impact on the total of the collection. Below is this list of characters, that could have a significant impact on the total of the collection.

```
“'”     Apostrophe      704
“-”     Hyphen          3856
“.”     Dot             25117
“,”     Comma           101537
```

The apostrophe character has 704 occurrences in the *individual* label fields. On closer observation one can easily see that the occurrences are not part of any notation scheme, except that they could be part of a name (example: *O'Shea, Tim*). This indicates that the characters are just some part of the label/name itself. The *individual* fields with *hyphen* characters, also has an occurrence which is significant (3856) in the test collection, but they are also not part of any notation that should be normalized.

A discovery made while looking at the *hyphen* occurrences was the use of parenthesis as a clarification of the information in the field. This lead to a full exploration of patterns in the fields

containing any kind of special character. The only pattern found was the pattern just mentioned. This pattern applied to parenthesis, curly brackets, square brackets, and angle brackets. Another feature of these bracket-patterns is that one of them is used as start of clarification, and a different can be used at the end (example below). Note that two of the fields containing brackets, are totally surrounded by them, and therefore it is not a clarification after all.

```
"Norske sivilingeniørers forening .[Kurs:) elektriske installasjoner på
sykehus”
```

The *dot* character occurred in 25117 *individual* fields. While looking through the *sub-fields* containing dots, one discovers that the dot is used only for shortage. These shorts could be either sub names or words. The particular words one is talking about here, is social or work titles, but most of these cases where found in a notation scheme of two commas. The occurrences where this type of word is not in conjunction with a *double comma* notation, indicates that it is actually part of the name itself, and not some notation of social or work title.

The *double comma* notation has already been revealed, but there are also more patterns among the 101537 comma fields individual names. The most common variance of individual name forms, is the *one comma* notation where the last name comes before the first comma. The other forms of an individual name contains more than one comma, and can be divided into five different categories. These categories are a *keystroke error*, a *individual with title*, a *individuals middle name initials* and *several individuals*. The several individuals category can also contain entries with less than two commas. Common to all the categories, is that the last name always comes before the first comma. The last category, several individuals, seems like the most complex record entries.

7.2.1.1 Keystroke Error

The most common keystroke error is a comma misplaced at the end of a person name field. This kind of error is easy to detect, and should be very easy to correct later on. There are 35 of these occurring in the record samples. An example of such a keystroke error is “Hemingway, Ernest,”.

7.2.1.2 Individual with Title

A title is an individual’s social or work title. Such person names with titles, are indicated by a second comma before the title itself.

```
Bowen, Richard LeBaron, Jr.
```

7.2.1.3 Individual Middle Name Initials

The entries in fields which go into the person middle name initials category, is those that contain the persons middle name initials after a second comma. There are eleven of these entries, an example of it is “Brook, Charles, G.D.“.

7.2.1.4 Several Individuals

The reason that the several persons category can be difficult to handle, is that there are more than one way of expressing several individuals. One example of two or more persons with the same last name, example; “Barlow, Jane and Michael Daly, Michael Noble and George Smith”. In the sample collection you can find seven individual fields where commas are used to separate different persons, and four instances where *and* is used as a separator between distinct individuals.

7.2.2 Conventions

Meetings, gatherings, events, workshops and so on can be called a convention. The convention class of fields in BIBSYS-MARC are in good conditions. But as for individuals, there are some usage of special characters (Table 3).

Character	UTF-8 Value	Occurrence
(	40	48
)	41	29
,	44	31
-	45	2878
.	46	310
/	47	11
;	59	1
<	60	1
>	62	1
?	63	43
[	91	1
]	93	1

Table 3: Special characters in time-frame fields

Two patterns where found in the convention fields, the first one was the *bracket pattern*, which was also found in individual fields. The second pattern, was a pattern associated with the comma character.

The *comma pattern* that was found, where indicating *geographic names*, *sequences*, or *dates*. *Geographic names* that do not belong to the convention name itself, are found behind one or more commas. But be aware that you can also find parts of the convention name after commas. *Dates* and *sequence numbers* appears anywhere in the field, but it seems like they could confuse the disambiguation process, for example by not matching to a convention of another year. In fact if the *time-frame* and *sequence numbers* where not incorporated in the name, it would be a relief to the developer. Here is a list of examples of what is mentioned above.

```
"Symposium on Mathematical Optimization Techniques, Santa Monica, California,
1960"
"Convegno di studi su Bartolomeo Ammannati-scultore e architetto,  1511-1592"
"Photography 1900: the Edinburgh Symposium"
"American Chemical Society Symposium on Chemistry, Structure and Reactivity
of Coals, Tar Sands and Oil Shale"
"Congress of Scandinavian psychiatrists 15"
```

There are 2490 records with convention fields in the sample collection. There was 91 entries with time-frames, 16 entries with sequence numbers, 14 geographic names and 26 bracket enclosures. The geographic names where; *Alghero, California, Cleveland, Ohio, Romain, Santa Monica, Sapporo, Tokyo, Møre og Romsdal, N.Trøndelag, Nordland, Troms* and *Finnmark.* Some of these was part of the label, while others should have been registered in a separate field.

7.2.3 Corporations

The use of special characters in corporation fields, are shown in the table below (Table 4). As for individuals and conventions, brackets and commas are associated with important patterns. In total there are 396 corporation entries with at least one comma in it.

Character	UTF-8 Value	Occurrence
!	33	1
“	34	38
&	38	95
‘	39	222
(	40	187
)	41	187
+	43	1
,	44	460
-	45	1167
.	46	684
/	47	128
:	58	17
;	59	4
<	60	10
>	62	9
?	63	54
[	91	2
]	93	1
‘	96	3
´	180	1

Table 4: Special characters in corporation labels

As for convention and individual names, there are brackets used for some information originally belonging to another field, in all 187. From a normalization point of view, the content of these can be ignored.

The four subjects that are brought up when looking at entries with commas are *geographic names*, *corporation types* and *dates*. In the sample collection you will find 135 *geographic names*, 12 *corporation types*, and 8 *dates*. The *corporation types* are; *AB, Aktiebolaget, Aktieselskabet, A/S, AS, Co., Forening, Group, I/S, Inc., Ltd.* and *Stiftelsen.*

Here are the geographical names; *Ahmedabad, Alma Ata, America, Amsterdam, Antwerp, Australia, Bad Godesberg, Bagdad, Balestrand, Barbados, Bardaw, Basel, Bath, Baton Rouge, Belgium, Berkeley, Berlin, Bergen, Bloomington, Barcelona, Boston, Bremen, Brussel, Caen, California, Casablanca, Ceylon, China, Copenhagen, Coral Gables, Croatia, Dresden, Dortmund, Drammen, Dublin, Durham, England, Essen, Finland, Firenze, France, Germany, Great Britain, Gorkij Zivopis, Guildford, Halmstad, Hannover, Helsinki, Høvikodden, Ireland, Italy, India, Japan,*

Karachi, Kartuzy, Kiel, København, Knoxville, Kunming, Leikanger, Lisboa, London, Los Angeles, Lund, Lusanne, Madrid, Malta, Manchester, Miami, Milde, Molde, Moscow, Munich, Münster, Møre og Romsdal, Nashville, Nebraska, New York, Nordland Fylke, Normandie, Norway, N.C, N.J, N.Y, New Brunswick, Omaha, Oslo, Otra, Pakistan, Paris, Poland, Portland, Posen, Reykjavik, Romania, Rome, Rovigno, Rungsted, Russia, San Francisco, Sandefjord, Santa Barbara, Seoul, Sogndal, Spain, St. Bonaventure, Stanford, Stavanger, Stockholm, StrasBourg, Surrey, Switzerland, Szombathely, Tallin, Tanzania, Telemark, Tokyo, Trondheim, Tunisia, Vaduz, Valletta, Varberg, Venice, Verdal, Vestfold, Volda, Walton, Warszawa, Washington, Washington D.C, Western Germany, Wien, Zagreb, Zürich and *Zyrich.*

The largest set of *corporate labels* are those not containing neither comma nor parenthesis (14276).

7.2.4 Publications

The patterns found in the special character set of the title fields (Table 5) are somewhat different from *individuals*, *conventions* and *corporations*. It is actually very varying with brackets, double quotes, single quotes, dates, paragraph references, commas an place names. But all these notations seem to be part of the publication label/title, except from;

- those that only contained dates
- those where the whole label where encapsulated in brackets.

There where 23 fields with only paragraph references, 243 fields with only dates, and 332 fields that where encapsulated by any form of brackets. In total there where 99110 *publication titles* in the test collection. Examples of the notations described are shown below.

- Paragraph reference: `§§ 830-838`
- Only Dates: `10/5-92`
- Encapsulated in brackets: `(Employment policy)`

Character	UTF-8 Value	Occurrence
!	33	247
“	34	1684
#	35	27
$	36	3
%	37	9
&	38	521
‘	39	4509
(	40	1485
)	41	1487
*	42	18
+	43	83
,	44	9616
-	45	13162
.	46	8024
/	47	1155
:	58	1137
;	59	1085
<	60	500
=	61	139
>	62	497
?	63	934
[	91	456
\	92	1
]	93	446
‘	96	40
{	123	1
|	124	3
}	125	1
~	126	1
§	167	76
¨	168	1
°	176	15
´	180	19

Table 5: Special characters in publication labels

8 Solution

The problem of automatically or semi-automatically disambiguate conventions, corporations and individuals that are responsible for some publication(s) is done by implementing a system names iDup. It was decided that iDup should be implemented totally in Java, but because of the search facility it was destined to provide, some parts (edit-distance similarity) of it also had to be implemented in C to meet the UDF[15] interface that the chosen database has (MySQL).

The communication between the Java system and the backend MySQL database is through the JDBC-Connector implemented specific to MySQL. The system architecture consists of modules implementing different interfaces. The application and the modules are initialized at start-up by providing a properties file containing information on what implementation of the different modules should be used. One module can access any other loaded module in the system through a *Central*. The *Central* has knowledge of all the loaded modules.

The system has seven distinct module interfaces; *Adapter*, *Disambiguater*, *LabelMeasurer*, *Loader*, *Log*, *Preprocessor* and *TimeframeMeasurer*. By implementing these interfaces to collaborate with each other, one achieves a functioning disambiguating system.

A small object model was created to ease the communication through the modules (especially through the *Adapter*, *Disambiguater* and *Loader* modules). The object module consists of *Row* and *Rows*, as the names suggest *Rows* wraps several *Row* instances. A row has seven fields/columns that is already familiar:

```
<row>
  <time-frame>29 Nov 1981</time-frame>
  <time-frame-preprocessed>29/11/1981:0/0/X</time-frame-preprocessed>
  <label>Myrhaug, Kristian Asle</label>
  <label-preprocessed>kristian asle myrhaug</label-preprocessed>
  <new-id>25idupOne-17891764621</new-id>
  <old-id></old-id>
  <record-number>25</record-number>
</row>
```

Time-frame, label and old id is the original information found in the MARC records. The time-frame and label fields are then processed and put into the corresponding preprocessed fields, while the old id field is copied into the new id field. The record number is a sequential number to separate the content of one record from the content of another. The record number starts at one and increases by one for each record that is loaded into the adapter.

15 User Defined Function

8.1 Interface Modules

8.1.1 Adapter

The adapter is a storage and search interface to a persistent storage[16]. The interface provides other modules with the capability to get rows that do not contain identifiers, get rows with a specified record number, get rows similar to another given row, insert rows, update rows, and get the highest record number. The get similar row function can be used at a later stage for entering new records, then the the system can give a choice of similar entities which a user can link the new record to. It can also be used to disambiguate automatically.

8.1.2 Disambiguater

A implementation of the Disambiguater uses the Adapter, LableMeasurer and TimeframeMeasurer to disambiguate the rows that can be accessed through an Adapter module (Illustration 4). The LabelMeasurer and TimeframeMeasurer are used to measure the distance or similarity between labels and time-frames. This distance or similarity should then be used to decide whether to entities match or not.

16 A persistent storage, could be a file system, a database or other storage that could be read or written to.

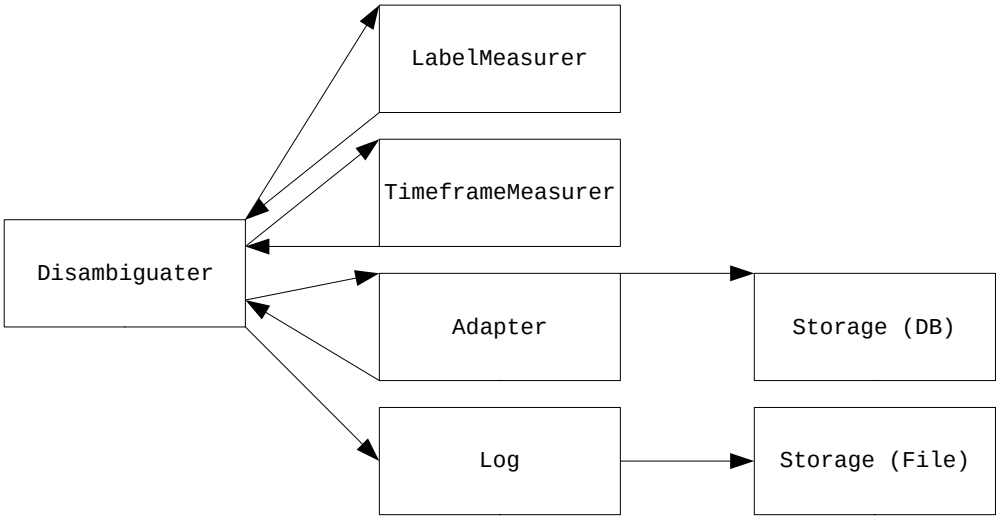

Illustration 4: Disambiguater using other modules to disambiguate records

8.1.3 LabelMeasurer and TimeframeMeasurer

A LabelMeasurer and TimeframeMeasurer should be able to calculate the distance or similarity between two values (label or time-frame). The implementation of the measurer's could vary, but the guideline for the similarity measure outcome is percent. The fact that one can easily replace a implementation of a module by editing a properties file, make the system very dynamical for testing different measures (edit-distance, time-frame distance, etc.).

8.1.4 Loader

A Loader implementation should load records into the system from a specific record source (MARC, MarcXchange, etc). The loader should use Preprocessor modules to fill a row with information (Illustration 5). A filled row should then be inserted into a storage, via a Adapter module.

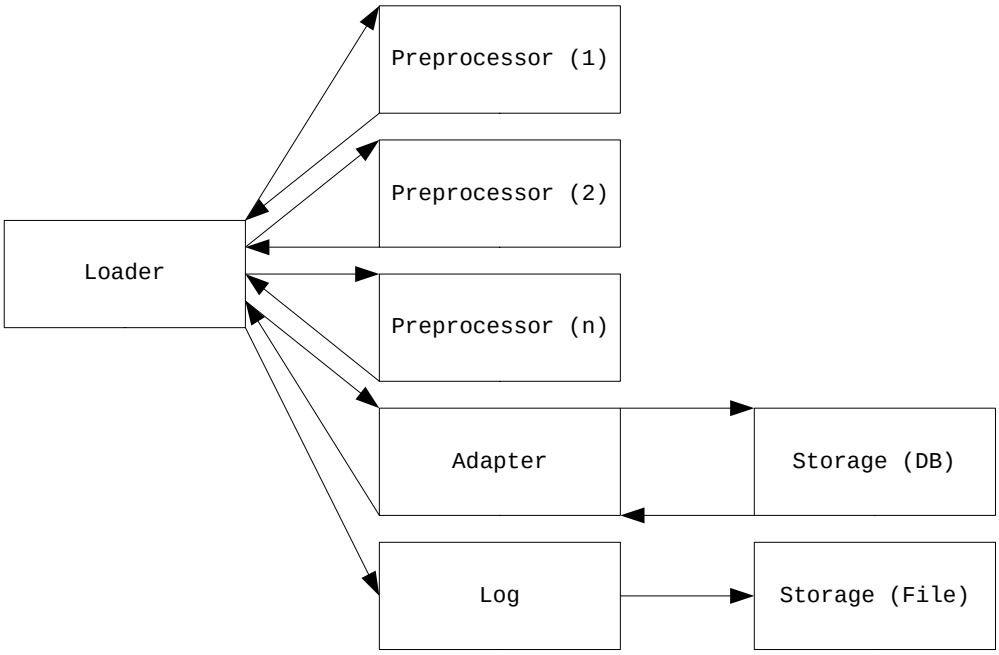

Illustration 5: Loader using other modules to load records

8.1.5 Log

A Log module should be used to track changes made by either a Disambiguater or Loader module. A Loader could for example use a Log module to log every row that was inserted into a storage. Another example would be a Disambiguater module logging all entity matches.

8.1.6 Preprocessor

There should be one implementation of a Preprocessor for each information type. For example an individual label and a corporation label should have separate preprocessors, because they are normalized in different ways. The time-frame field should also have a dedicated preprocessor. As mentioned the preprocessor should be used by a Loader to accomplish a normalized form of each field.

8.2 Implemented Modules

All the interface modules, *Adapter*, *Disambiguater*, *LabelMeasurer*, *Loader*, *Log*, *Preprocessor* and *TimeframeMeasurer*, had to be implemented to achieve test results. As mentioned before these should be implemented to collaborate with each other to achieve the disambiguation and the loading tasks. The implementations of the different models are listed below.

Module Interface	Implemented Modules
Adapter	JDBCAdapter
Disambiguater	Disambiguater1
LabelMeasurer	LevenshteinMea
Loader	MarcXchangeLoader
Log	StandardLog
Preprocessor	ConventionPre CorporationPre IndividualPre IdentifierPre PublicationPre TimeframePre
TimeframeMeasurer	TimeframeMea

8.2.1 JDBCAdapter

A JDBC connector implements the *java.sql* interfaces to provide SQL access to a database. The *JDBCAdapter* provides a more convenient interface, between other modules and the selected JDBC connector. The database selected in the current implementation is MySQL, and the MySQL JDBC connector is therefor set in the *Central's* properties file.

The distance measurer concerned with names had to be implemented both on the database-side and the application-side of the system. The database-side edit distance measurer is in C, adapted to the User Defined Function interface that MySQL provides. The reason for the database-side implementation is a retrieval issue of entities that are possible matches.

8.2.2 Disambiguater1

The disambiguation process is controlled by the *Disambiguater1* module. When disambiguating a specified entity type, for example *individuals*, the first action is to fetch the next row without a new-id assigned to it (using the current Adapter implementation). When this initial row is fetched, the process continues by searching for rows satisfying a similarity tolerance between the initial preprocessed label and a row's preprocessed label. Finally a match is confirmed by either a time-

frame or publication label within a specified similarity tolerance. If there were no matches with rows with an assigned id, an id is generated for the rows representing the same entity. This process continues until there are no more rows without a new-id.

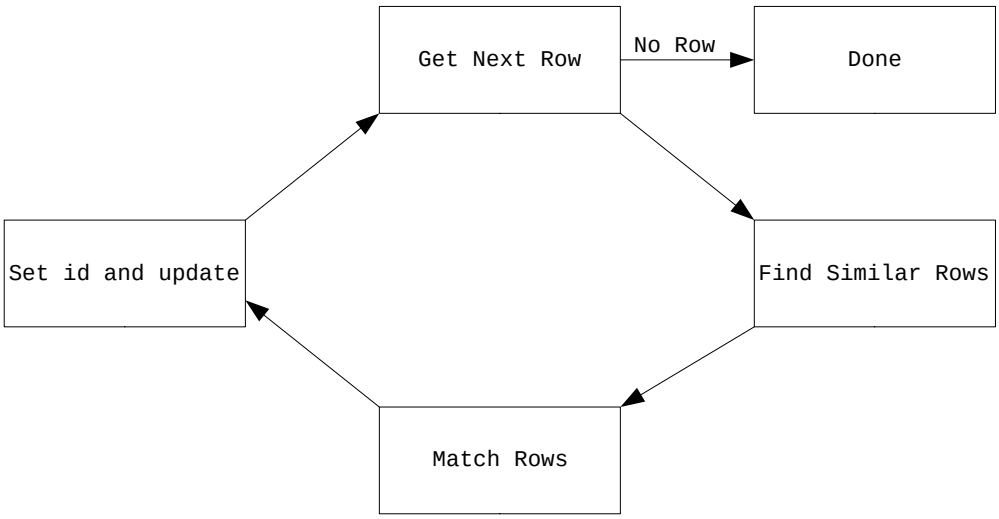

Illustration 6: Disambiguating process

Finding next row and rows similar to it is the Adapter module's task. But deciding whether two rows match with each other is done by exploring the similar rows' time-frames and publication labels.

When matching the similar rows to the initial row the *Disambiguater1* is building an extended entity, by remembering all matching rows, including their publications. By traversing the similar rows, and matching them against the extended entity, until no more matches are found one builds this extended entity. Since this extended entity contains all rows that are decided to be the same entity, one just updates these rows with a new-id (if they do not have one already). Off course this new-id is selected from one of the rows containing an old-id. If there is several old-ids, the first encountered row is selected to provide the identifier. There is also the case where there is no existing old-id among the extended entity rows, in that case a id is generated.

The generation of ids depend on whether there is only one row in the extended entity, or several. When there is one row, this is annotated by “iDupOne” in contrast to “iDupMore”, when there are

several rows. Further the initial row's record number and a hash code of the initial row's label are also part of the generated id, like this; record-number + [iDupOne | iDupMore] + hash code. Here are two examples:

```
133iDupOne2097808535
624iDupMore-1359411214
```

8.2.3 LevenshteinMea

The implementations of the edit distance algorithm involves a matrix of size (m+1) x (n+1), where m is the length of the first string, and n is the length of the second string. If one of the strings have the length of 0, the other strings length is the distance. The first row and first column are filled with integers starting at 0 . The rest of the matrix are calculated by the minimum of the nearest north + 1,nearest north-west + cost, and nearest west + 1 value. The cost is 0 if the characters corresponding to the matrix position are equal, and 1 otherwise. You will find the edit distance in the bottom-right of the matrix.

		K	R	I	S	T	I	A	N
	0	1	2	3	4	5	6	7	8
M	1	1	1	3	4	5	6	7	8
Y	2	2	2	3	4	5	6	7	8
R	3	3	2	3	4	5	6	7	8
H	4	4	3	3	4	5	6	7	8
A	5	5	4	4	4	5	6	6	7
U	6	6	5	5	5	5	6	7	7
G	7	7	6	6	6	6	6	7	8

The similarity of to labels is the percent of correct characters relative to the longest text string.

8.2.4 MarcXchangeLoader

As mentioned MarcXchange is a XML scheme that supports exchange of different coded MARC formats between two or more bibliographic management systems. The iDup system was presented with such a exchange format. The records from the XML files had to be loaded into a persistent storage, because the quantity of the records where to large to keep in memory/ram at run time.

As explained before, individuals are loaded from 100, 600 and 700, corporations from 110, 610 and 710, conventions from 111, 611 and 711, and finally titles from 240, 241 and 245 record fields. Names are found in the *sub-field* $a, dates in $d, and old identifiers in $5. This is configurable in the *Central*'s properties file.

When loading, all preprocessed labels that are equal in a specific record are merged into one row, to minimize the table size. This merger is not applied if there is a miss-match in either the preprocessed time-frame or the new-id fields. The common factor connecting content of one record across the tables are a sequential number given when a record is loaded into the database, called record-number, the number starts at 1 and increases by one for each record that is loaded.

8.2.5 StandardLog

The standard log is rather simple, it just writes whatever is sent to it into a file. The file name is retrieved from the *Central*'s properties file.

8.2.6 ConventionPre

A convention name/label goes through a filter that ignores character different from digits, letters (except a single quote inside a sub name). Other characters following each other are narrowed down to one space, and everything enclosed by brackets are ignored. Here is an example from convention labels:

```
Before:
Industriseminarium (7 sept. 1993 : Stockholm)
After:
industriseminarium
```

8.2.7 CorporationPre

The understanding of conventions and corporations are closely related to each person's perception of what they are, and you can find conventions in corporation fields and visa versa. For this reason, the convention and corporation labels are preprocessed in the same way. Here is an example from corporation labels:

```
Before:
Centre d'études catalanes (Paris, France)
After:
centre d'études catalanes
```

8.2.8 IdentifierPre

This *Preprocessor* is more like a dummy module, it just returns the original identifier. The reason for having this module at all, is that it should be possible to map the loaded entity's identifier to another identification scheme.

8.2.9 IndividualPre

Preprocessing individual labels starts in the same manner as the convention and corporation preprocessors. The crucial difference is in the handling of commas. Splitting the label in three

sections, the first section is everything before a first comma, the second section is everything between the first and the second comma, and the third section is everything after a second comma.

The first and the second section is swapped with each other[17], and the third section is ignored, because of the five double-comma categories found while examining the content of BIBSYS.

```
Before:
Roll, Charles Robert, jr.
After:
charles robert roll
```

8.2.10 PublicationPre

Preprocessing publication titles/labels does the same as convention and corporation preprocessor, except from ignoring the content of enclosing brackets.

```
Before:
Robert Burton (1577-1640) et "L'anatomie de la melancolie"
After:
robert burton 1577 1640 et l'anatomie de la melancolie
```

8.2.11 TimeframePre

Based on the observations made, a proposal of preprocessing the date field has been made. Here is a walk-through of how the proposal for automatically reducing most of the different formats into a standard time-frame format of the form *day/month/year:day/month/year*, where the first date is the start point and the second date is the end point.

- Days can have values from 0 through 31
- Months can have *0, January, February, March, April, May, June, July, August, September, October, November* and *December*
- Years can take *X* and all positive and negative integers.

The 0 values of days and months, and the X value of years specify that the value is undefined. This allows for very accurate dates or very inaccurate dates, which is just the type of information that time-frame fields in BIBSYS contain. The preprocessors task is to change the input into the described format. To show how this is done, it is proper to “walk” through the process, by preprocessing four examples:

```
“ca. 4 B.C.-30/1 A.D.”
”d. Nov 29, 1981”
”29.-11 30.-11, 1981”
”11 28.- 29.- 30.-, 1981”
```

17 Swapped because of surname comes before a first comma.

The first step of the process is to strip everything but letters and digits. While the striping process is running other tasks are also executed;

- count up possible days and possible years
- finding guaranteed months
- replacing “/” with “or”
- replacing “cent” with “century”
- mapping down to one language.
- removing number suffixes (e.g. st, nd, rd, th).

The stripping process gives us a more manageable state of the field content.

```
“ca 4 B C 30 or 1 A D (d:3, y:0)”
”d 29 1981 (d:1, y:1, m1:November)”
”29 11 30 11 1981 (d:4, y:1)”
”11 28 29 30 1981 (d:4, y:1)”
```

After stripping the fields, it is time for annotation scouting. When a “b.c” is found, it is annotated as “Before Christ” and every integer on its left gets a negative value.

An “a.d” notation is ignored if no *Before Christ* is present. If a *Before Christ* notation is present every integer between the *Anno Domini* and the *Before Christ* notation are perceived as a year, because it is believed that day and months are not known for this very old information source.

The “d” notation is replaced by a “deceased” notation, and the next integer after an “or” is ignored - that is if “century” is not present. The “ca” notation is just ignored/forgotten, because the proposed format does not support it.

```
“-4 30 (d:0, y:2)”
”28 1981 (d:1, y:1, m1:November, deceased)”
”29 11 30 11 1981 (d:4, y:1)”
11 28 29 30 1981 (d:4, y:1)”
```

Now comes the tricky part of placing these number in the correct position. Starting with years;

- if the “century” notation is present, the start-date year is set to the lowest in this time-frame, and end-date year is set to the highest year in this time-frame.
- if year-count is larger than zero, then pick the leftmost year as start.
- if year-count is larger than one, pick the rightmost year as end (and remove the other years).

The outcome of this is as follows.

```
“(d:0, y:0)”
  0/0/-4:0/0/30
“28 (d:1, y:1, death)”
  0/November/1981:0/0/X
“29 11 30 11 (d:4, y:0)”
  0/0/1981:0/0/X
“11 28 29 30 (d:4, y:0)”
  0/0/1981:0/0/X
```

Further on, one tries to fill in the blank fields by using the knowledge of the remaining days. By parting the handle method into those already containing the start-month, and those that do not, the complexity is narrowed.

8.2.11.1 Containing start month

There are four possible outcomes of dates already containing start month. To find the right outcome one has to use the knowledge of remaining days.

One day

Insert the possible day integer into the start date day.

Two days

Insert the first of the remaining days into the start-day, and insert the other day into the end-day field.

Three days

Insert the first of the remaining days into start-date day field, and insert the last of remaining days into the end-day.

Four days

Insert the first of the remaining days into start-day field, and insert the last of remaining days into the end -date day.

8.2.11.2 Not containing start month

When the start-month is not filled in the outcomes can be separated into four categories, depending on the remaining day-count.

One day

Insert the possible day integer into the start month field.

Two days

First check if d2 is a month, if it is, insert the corresponding month into start date month, and d1 into start date day. Otherwise, if the d1 field is a month, do the opposite. If non- of this kick in nothing is done.

Three days

First check if d3 can be a month, if it is, insert the month into start-date month, and d1 into start date day, and d2 into end date day - Otherwise, check if d1 can be month, if so, insert the this month into start date month, and d2 into start date day, and d3 into end date day - Otherwise, check if d2 can be month, if this is true, insert the this month into start date month, and d1 into start date day, and d3 into end date day.

Four days

First check if d2 and d4 can be months, if so, insert d2 into start date month, d4 into end date month, d1 into start date day, d3 into end date day -Otherwise, check if d1 and d2 can be months, if so, insert d1 into start date month, d3 into end date month, d2 into start date day, d4 into end date day. Otherwise check if d4 can be a month, if this is true, insert d4 into start date month, d1 into start date day, and d3 into end date day.

Here is the result of applying the above to the examples we are following:

```
0/0/-4:0/0/30
28/November/1981:0/0/X
29/November/1981:30/November/X
28/November/1981:30/0/X
```

8.2.11.3 Cleaning step

The final step in the process is to clean the result of preprocessing a date field. This involves filling out missing parts. Like for instance, when there is a end date month present, but not an end date year present, one should copy the start date year into the end date year.

```
0/0/-4:0/0/30
28/November/1981:0/0/X
29/November/1981:30/November/1981
28/November/1981:30/November/1981
```

8.2.12 TimeframeMea

Time-frames are an important information source because it can be of help while filtering, or in this

case, confirming/strengthening the relevance of different information types. As discovered in the observation chapter, the variety of formats in the fields make it difficult to reduce it to a specific form. The time-frame format described in the TimeframePre, is here subject to distance and similarity measuring.

```
day/month/year:day/month/year
```

The distance between time-frames have a range between 0 and infinite. Before we look at how two measure the distance between time-frames, we need to look at how the distance of two dates are found.

When measuring the distance between two dates, the basis is the date with least certainty. This date is the date with the most unspecified parts (day, month, year).

- If the least certain date is unspecified in all parts, or in the year part, the distance is infinite.
- If the least certain only has the year specified, the difference of the dates' year is the distance.
- If the least certain date has the month specified, the difference in the dates' month is the distance, unless the dates' year are differing. When the years are differing the distance is infinite.
- If the least certain date has the day specified, the difference in the dates' day is the distance, unless the dates' year or month are differing. When the years or the months are differing the distance is infinite.

Now, let us look at the time-frame distance. When both start-dates and end-dates has been measured, they are added together to make up the time-frame distance, except when exactly one of the distances are infinite, then the time-frame distance is the definite date distance.

The similarity of time-frames is set to 0% if the distance is 20 and 100% if distance is 0 years.

9 Evaluation and Discussion

To evaluate the impact of the system, there has to be something to compare it with. The BIBSYS content provides proprietary identifiers, which indicate that it is the internal authority control. If iDup is able to find more entries (true positive) of one or several identities, it is superior to any other disambiguation application[18] which BIBSYS has used in the past. The disambiguation of a library catalogue can not allow for false positive disambiguation. The iDup system will therefore fail if there is any of these. The true negative set are to large to explore manually, but on a Levenshtein edit-distance similarity of 100% there would probably be several of these, for example:

```
Wolfgang Amadeus Mozart
Wolfgang A. Mozart
```

To see whether iDup, set up with the Levenshtein edit-distance similarity, is capable of matching some abbreviations of entities and discriminate non-ambiguous entries, it is set up to bring abbreviating entities together by running the system three times, with three different settings. The chosen sample collection is 50000 records, which is a subset of the previously explored collection.

The first run was the control run, to see if everything was working properly and maybe find some matches. The similarity tolerance on the control run was set to 100% on *individual labels*, *publication labels* and *time-frames*. The time-frame similarity is 100% on distance 0 and 0% on distance 20.

The second run had *labels* and *time-frame* similarity tolerance set to 90%. This is probably a harsh choice of values, but it is meant to show that the measures tolerates some fault/abbreviation, but is still able to discriminate non-ambiguous labels and time-frames.

The third run had tolerance values that should be considered as too forgiving; *Label* and *time-frame* similarity tolerance is 50%. The hope in this run is actually to see the system disambiguating non-ambiguous entities.

9.1 The 100% Run

The results of the first run showed that the dynamic system architecture was functioning well, that the process was time consuming, and that the preprocessing of fields added value to the system. The updates made to entries in the database where logged into an XML file, which could be viewed after a run. To evaluate the result, reviewing the log had to be done. The log showed that there was 41 entity matches with the BIBSYS proprietary identifiers, and 109 entity matches within the set of rows that had no previous identifier assigned to it. The run found 13225 singletons that did not match any other entity.

18 Manually, semi-automatically or automatically disambiguation.

9.1.1 Proprietary Identifier Matches

The 41 entities (150 entries involved) that was disambiguated, and updated with a proprietary identifier where confirmed by either a *time-frame*, a *publication label* or both. The system log showed the match category that had occurred. The matches within the proprietary identifier set, had matches in all three categories. We should look at some samples of these matches found in the log.

9.1.1.1 Time Frame Match

Here is a sample of a *time-frame* match. A look at the *timeframe* and *timeframepre* attribute confirms the match and shows us the value of preprocessing *time-frame* fields:

```
<merge>
  <individual date="ca. 4 f.Kr.-65 e.Kr." datepre="0/0/-4:0/0/65" label="Seneca, Lucius Annaeus" labelpre="lucius annaeus seneca" new_id="x90585815"
sequence="34050"/>
  <individual date="ca. 4 B.C.-65 A.D" datepre="0/0/-4:0/0/65" label="Seneca, Lucius Annaeus" labelpre="lucius annaeus seneca" new_id="x90585815"
sequence="41570"/>
  <individual date="ca. 4 f.Kr.-65 e.Kr." datepre="0/0/-4:0/0/65" label="Seneca, Lucius Annaeus" labelpre="lucius annaeus seneca" new_id="x90585815"
old_id="x90585815" sequence="4229"/>
  <individual date="ca. 4 f.Kr.-65 e.Kr." datepre="0/0/-4:0/0/65" label="Seneca, Lucius Annaeus" labelpre="lucius annaeus seneca" new_id="x90585815"
old_id="x90585815" sequence="28723"/>
</merge>
```

9.1.1.2 Publication Label Match

Here is an example of label match. The consistency was confirmed by a lookup in the *publication* table at the respective *record numbers:*

```
<merge>
  <individual label="Wright, A" labelpre="a wright" new_id="x90334464" recordnum="9005"/>
  <individual label="Wright, A." labelpre="a wright" new_id="x90334464" old_id="x90334464" recordnum="18082"/>
</merge>
<publications>
  <publication label=”Electrical power system protection” labelpre=”electrical power system protection” recordnum=”9005”/>
  <publication label=”Electrical power system protection” labelpre=”electrical power system protection” recordnum=”18082”/>
<publications/>
```

9.1.1.3 Time Frame and Publication Label Match

Sample of *time-frame* and publication match, we see that the individual in record number 17602 links the rest of the matches to a BIBSYS proprietary identifier from record number 49735:

```
<merge>
  <individual timeframe="1564-1616" timeframepre="0/0/1564:0/0/1616" label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737"
recordnum="2502"/>
  <individual timeframe="1564-1616" timeframepre="0/0/1564:0/0/1616" label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737"
recordnum="13931"/>
  <individual timeframe="1564-1616" timeframepre="0/0/1564:0/0/1616" label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737"
recordnum="13951"/>
  <individual timeframe="1564-1616" timeframepre="0/0/1564:0/0/1616" label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737"
recordnum="16144"/>
  <individual timeframe="1564-1616" timeframepre="0/0/1564:0/0/1616" label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737"
recordnum="18488"/>
  <individual timeframe="1564-1616" timeframepre="0/0/1564:0/0/1616" label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737"
recordnum="35246"/>
  <individual timeframe="1564-1616" timeframepre="0/0/1564:0/0/1616" label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737"
recordnum="35294"/>
  <individual timeframe="1564-1616" timeframepre="0/0/1564:0/0/1616" label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737"
old_id="x90052737" recordnum="17609"/>
  <individual label="Shakespeare, William" labelpre="william shakespeare" new_id="x90052737" old_id="x90052737" recordnum="49735"/>
</merge>
<publications>
```

```
  <publication label=”Macbeth” labelpre=”macbeth” recordnum=”17602”/>
  <publication label=”Macbeth” labelpre=”macbeth” recordnum=”49735”/>
 </publications>
```

9.1.2 New Identifier Matches

There was 109 entities that had more than one entry (240 entries in total) within the set of entries that has no proprietary identifier. The fact that there was more matches within this set is not surprising, since the person who put them into a BIBSYS catalogue probably was not able to find a possible match in authority controlled set, and therefore added it without an authority control identifier. This would probably have happened several times - hence there are more entities with several entries within the set with no proprietary identifier.

9.1.2.1 Time Frame Match

This sample shows that the *time-frame* similarity is working properly, by confirming that `0/0/X:0/0/1400` is within the scope of `0/0/1345:0/0/1400`:

```
 <more>
  <individual timeframe="1345-1400" timeframepre="0/0/1345:0/0/1400" label="Chaucer, Geoffrey" labelpre="geoffrey chaucer" new_id="14777iDupMore-1870756288" sequence="14777"/>
  <individual timeframe="1345-1400" timeframepre="0/0/1345:0/0/1400" label="Chaucer, Geoffrey" labelpre="geoffrey chaucer" new_id="14777iDupMore-1870756288" sequence="17592"/>
  <individual timeframe="d. 1400" timeframepre="0/0/X:0/0/1400" label="Chaucer, Geoffrey" labelpre="geoffrey chaucer" new_id="14777iDupMore-1870756288" sequence="23503"/>
 </more>
```

9.1.2.2 Publication Label Match

These two samples shows evidence of co-authorship matches, which is a great feature of iDup:

```
 <more>
  <individual label="Mandeville, John" labelpre="john mandeville" new_id="1824iDupMore-2050173866" recordnum="1824"/>
  <individual label="Mandeville, John" labelpre="john mandeville" new_id="1824iDupMore-2050173866" recordnum="1825"/>
  <individual label="Mandeville, John" labelpre="john mandeville" new_id="1824iDupMore-2050173866" recordnum="3139"/>
 </more>
 <more>
  <individual label="Halliwell, J.O." labelpre="j o halliwell" new_id="1824iDupMore1224928981" recordnum="1824"/>
  <individual label="Halliwell, J.O." labelpre="j o halliwell" new_id="1824iDupMore1224928981" recordnum="1825"/>
 </more>
 <publications>
  <publication label=”Itinerarium” labelpre=”itinerarium” recordnum=”1824”/>
  <publication label=”The voiage and travaile of Sir John Maundevile, kt” labelpre=”the voiage and travaile of sir john maundevile kt” recordnum=”1824”/>
  <publication label=”Itinerarium” labelpre=”itinerarium” recordnum=”1825”/>
  <publication label=”The voiage and travaile of Sir John Maundevile, kt” labelpre=”the voiage and travaile of sir john maundevile kt” recordnum=”1825”/>
  <publication label=”Itinerarium” labelpre=”itinerarium”/ recordnum=”3139”>
 </publications>
```

9.1.2.3 Time Frame and Publication Label Match

There was no entities updated with a new identifier.

9.2 The 90% Run

The 90 % run confirmed that the system was able to allow small errors in labels, and thereby be able to disambiguate more entities than the non error tolerance 100% run. But it also has one false

positive match, which bring the 90% setting to be a failure:

```
<merge>
  <individual label="Metcalf, C.L." labelpre="c l metcalf" new_id="x90374162" recordnum="12545"/>
  <individual label="Metcalf, R.L." labelpre="r l metcalf" new_id="x90374162" old_id="x90374162" recordnum="12545"/>
</merge>
<publications>
  <publication label=”Destructive and useful insects” labelpre=”destructive and useful insects” recordnum=”12545”/>
</publications>
```

An immediate improvement to this, could be to have ask an human operator whether a match of less than 100% accuracy really is a true or false match. This is close to the categorisation of *Record Linkage* (section 4.2.1). Another improvement to this would be to change the measure to a combination of edit-distance and *vector space* (section 3.4.2.3) similarity.

9.2.1 Proprietary Identifier Matches

54 entity matches with a proprietary identifier, was reported in the log file. Many of these where true positive approximate matches, this means that the labels or *time-frame* where below 100% and above or equal to 90% accurate.

9.2.1.1 Time Frame Match

Sample from log, where a small suffix abbreviation was allowed:

```
<merge>
  <convention timeframe="1987" timeframepre="0/0/1987:0/0/X" label="Kulturlederkonferanse" labelpre="kulturlederkonferanse" new_id="x90637192"
recordnum="13986"/>
  <convention timeframe="1987" timeframepre="0/0/1987:0/0/X" label="Kulturlederkonferansen" labelpre="kulturlederkonferansen" new_id="x90637192"
old_id="x90637192" recordnum="46498"/>
</merge>
```

9.2.1.2 Publication Label Match

Here is a sample where the publication label confirmed a match, the individual has to different spelling (`Haxthow, Victor` and `Haxthow, Viktor`):

```
  <merge>
  <individual label="Haxthow, Victor" labelpre="victor haxthow" new_id="x90124262" sequence="47937"/>
  <individual label="Haxthow, Viktor" labelpre="viktor haxthow" new_id="x90124262" old_id="x90124262" sequence="32535"/>
</merge>
<publications>
  <publication label="Sjøfartsloven (1893)" labelpre="sjøfartsloven 1893" recordnum="32535"/>
  <publication label="Sjøfartsloven (1893)" labelpre="sjøfartsloven 1893" recordnum="47937"/>
</publications>
```

9.2.1.3 Time Frame and Publication Label Match

Sample where both *time-frame* and label where involved in the match:

```
<merge>
  <individual timeframe="1639-1699" timeframepre="0/0/1639:0/0/1699" label="Racine, Jean" labelpre="jean racine" new_id="x90052151" sequence="10532"/>
  <individual timeframe="1639-1699" timeframepre="0/0/1639:0/0/1699" label="Racine, Jean" labelpre="jean racine" new_id="x90052151" sequence="35444"/>
  <individual label="Racine, Jean" labelpre="jean racine" new_id="x90052151" old_id="x90052151" sequence="38724"/>
</merge>
<publications>
  <publication label=”Jean Racine - dramatisk” labelpre=”jean racine dramatisk” recordnum=”10532”/>
  <publication label=”Jean Racine” labelpre=”jean racine” recordnum=”35444”/>
  <publication label=”Jean Racine” labelpre=”jean racine” recordnum=”38724”/>
```

```
</publications>
```

9.2.2 New Identifier Matches

On the 90 % run, 133 entities where disambiguated, and the entries of these entities where updated with a new identifier.

9.2.2.1 Time Frame Match

A sample of *time-frame* match where the individual name/label are almost equal:

```
<more>
  <individual timeframe="1891-" timeframepre="0/0/1891:0/0/X" label="Sizzo-Norris, Margot" labelpre="margot sizzo norris" new_id="6124iDupMore-247112671"
recordnum="6124"/>
  <individual timeframe="1891-" timeframepre="0/0/1891:0/0/X" label="Sizzo-Noris, Margot" labelpre="margot sizzo noris" new_id="6124iDupMore-247112671"
recordnum="6124"/>
</more>
```

9.2.2.2 Publication Label Match

A sample of publication match where both entries are from the same record, but abbreviating. These entries are just as important to disambiguate because other abbreviations from another record, may match be within the accepted tolerance when it is not within the range of the other:

```
<more>
  <individual timeframe="ed." label="Shibanov, G.P." labelpre="g p shibanov" new_id="28939iDupMore1182974203" recordnum="28939"/>
  <individual timeframe="red." label="Sibanov, G.P." labelpre="g p sibanov" new_id="28939iDupMore1182974203" recordnum="28939"/>
</more>
<publications>
  <publication label=”Monitoring the functioning of large systems” labelpre=”monitoring the functioning of large systems” recordnum=”28939”/>
</publications>
```

9.2.2.3 Time Frame and Publication Label Match

There were no matches where both *time-frame* and publication label was involved in confirming a match.

9.3 The 50% Run

The 50 % run was expected to fail the discrimination of non-ambiguous entities, and it did on several occasions.

9.3.1 Proprietary Identifier Matches

9.3.1.1 Time Frame Match

Sample from log that shows a false positive *time-frame* match:

```
<merge>
  <individual timeframe="1943" timeframepre="0/0/1943:0/0/X" label="Albert, Kenneth J." labelpre="kenneth j albert" new_id="x97024377" sequence="1746"/>
  <individual timeframe="1945" timeframepre="0/0/1945:0/0/X" label="Hardy, Kenneth A." labelpre="kenneth a hardy" new_id="x97024377" sequence="9403"/>
  <individual timeframe="1947" timeframepre="0/0/1947:0/0/X" label="Seeskin, Kenneth" labelpre="kenneth seeskin" new_id="x97024377" sequence="41238"/>
  <individual timeframe="1950" timeframepre="0/0/1950:0/0/X" label="Bleakly, Kenneth D." labelpre="kenneth d bleakly" new_id="x97024377" old_id="x97024377"
sequence="23629"/>
```

```
</merge>
```

9.3.1.2 Publication Label Match

This sample shows a false positive publication label match:

```
<merge>
  <individual label="Hansen, Thorkild" labelpre="thorkild hansen" new_id="x90186114" sequence="4016"/>
  <individual label="Janssen, Horst" labelpre="horst janssen" new_id="x90186114" old_id="x90186114" sequence="102"/>
</merge>
<publications>
  <publication label=”Horst Janssen” labelpre=”horst janssen” recordnum=”102”/>
  <publication label=”Thorkild Hansen” labelpre=”thorkild hansen” recordnum=”4016”/>
</publications>
```

9.3.1.3 Time Frame and Publication Label Match

This shows a false positive match where both *time-frame* and publication label are involved.

```
  <merge>
   <individual timeframe="1901" timeframepre="0/0/1901:0/0/X" label="Lacan, Jacques" labelpre="jacques lacan" new_id="x90064827" sequence="2124"/>
   <individual timeframe="1910" timeframepre="0/0/1910:0/0/X" label="Guillon, Jacques" labelpre="jacques guillon" new_id="x90064827" sequence="28290"/>
   <individual timeframe="1895" timeframepre="0/0/1895:0/0/X" label="Madaule, Jacques" labelpre="jacques madaule" new_id="x90064827" sequence="38853"/>
   <individual label="Lacan, Jacques" labelpre="jacques lacan" new_id="x90064827" old_id="x90064827" sequence="39648"/>
  </merge>
  <publications>
   <publication label=”Lacan, de l'équivoque à l'impasse” labelpre=”lacan de l équivoque à l impasse” recordnum=”2124”/>
   <publication label=”” labelpre=”” recordnum=””/>
   <publication label=”Lacan ; de l'équivoque à l'impasse” labelpre=”lacan de l équivoque à l impasse” recordnum=”39648”/>
  </publications>
```

9.3.2 New Identifier Matches

9.3.2.1 Time Frame Match

Yet another sample of a false positive *time-frame* match.

```
  <more>
    <individual timeframe="1901-1944" timeframepre="0/0/1901:0/0/1944" label="Moen, Petter" labelpre="petter moen" new_id="1149iDupMore-71424991"
recordnum="1149"/>
    <individual timeframe="1900" timeframepre="0/0/1900:0/0/X" label="Simon, Werner" labelpre="werner simon" new_id="1149iDupMore-71424991"
recordnum="43340"/>
  </more>
```

9.3.2.2 Publication Label Match

A publication label false positive match:

```
  <more>
    <individual timeframe="1937" timeframepre="0/0/1937:0/0/X" label="Hayashima, Akira" labelpre="akira hayashima" new_id="2393iDupMore-1847495437"
recordnum="2393"/>
    <individual timeframe="1929" timeframepre="0/0/1929:0/0/X" label="Nagazumi, Akira" labelpre="akira nagazumi" new_id="2393iDupMore-1847495437"
recordnum="39043"/>
  </more>
  <publications>
    <publication label=”Die Illusion des Sonderfriedens” labelpre=”die illusion des sonderfriedens” recordnum=”2393”/>
    <publication label=”The dawn of Indonesian nationalism” labelpre=”the dawn of indonesian nationalism” recordnum=”39043”/>
  </publications>
```

9.3.2.3 Time Frame and Publication Label Match

A sample of false positive match by both *time-frame* and publication label:

```
<more>
  <individual timeframe="1345-1400" timeframepre="0/0/1345:0/0/1400" label="Chaucer, Geoffrey" labelpre="geoffrey chaucer" new_id="14777iDupMore-1870756288" sequence="14777"/>
  <individual timeframe="1345-1400" timeframepre="0/0/1345:0/0/1400" label="Chaucer, Geoffrey" labelpre="geoffrey chaucer" new_id="14777iDupMore-1870756288" sequence="17592"/>
  <individual timeframe="d. 1400" timeframepre="0/0/X:0/0/1400" label="Chaucer, Geoffrey" labelpre="geoffrey chaucer" new_id="14777iDupMore-1870756288" sequence="23503"/>
  <individual label="Savidge, Geoffrey" labelpre="geoffrey savidge" new_id="14777iDupMore-1870756288" sequence="45572"/>
</more>
<publications>
  <publication label=”Studies in Troilus” labelpre=”studies in troilus” recordnum=”17592”/>
  <publication label=”Studies of factor VIII” labelpre=”studies of factor viii” recordnum=”45572”/>
/publications>
```

9.3.2.4 Worst case

The worst case of false positive match in the log file, matched with several identities from the proprietary identification scheme. The system chose the first entry with a BIBSYS identifier for those that did not have any identifiers. The system should rather have given a new identifier, than an proprietary identifier, to separate them from the proprietary entries. The biggest reason for the perception that all of these entries are one entity is mainly the fact that time-frame similarity is set to allow 10 years of positive or negative offset from any other time-frame in the extended entity. The result of this offset acceptance can for example be 1940 is in the scope of 1931 which is in the scope of 1926, as we see this actually result in a larger time-frame acceptance than predicted. Maybe one should only allow a time-frame similarity tolerance/offset from the initial entry. Record number 1168 is the initial row that caused the collection of entries.

```
<merge>
  <individual timeframe="1926" timeframepre="0/0/1926:0/0/X" label="Shelton, Robert" labelpre="robert shelton" new_id="x97031807" sequence="1168"/>
  <individual label="Skelton, Barbara" labelpre="barbara skelton" new_id="x97031807" sequence="3944"/>
  <individual timeframe="1933" timeframepre="0/0/1933:0/0/X" label="Ruben, Robert J." labelpre="robert j ruben" new_id="x97031807" sequence="12116"/>
  <individual timeframe="1929" timeframepre="0/0/1929:0/0/X" label="Jordan, Robert S." labelpre="robert s jordan" new_id="x97031807" sequence="15832"/>
  <individual timeframe="1914" timeframepre="0/0/1914:0/0/X" label="Roelofs, Robert T." labelpre="robert t roelofs" new_id="x97031807" sequence="19046"/>
  <individual timeframe="1923" timeframepre="0/0/1923:0/0/X" label="Soulat, Robert" labelpre="robert soulat" new_id="x97031807" sequence="22474"/>
  <individual timeframe="1942" timeframepre="0/0/1942:0/0/X" label="Dirks, Robert" labelpre="robert dirks" new_id="x97031807" sequence="25011"/>
  <individual timeframe="1936" timeframepre="0/0/1936:0/0/X" label="Sadoff, Robert L." labelpre="robert l sadoff" new_id="x97031807" sequence="28108"/>
  <individual timeframe="1952" timeframepre="0/0/1952:0/0/X" label="Hill, Robert M." labelpre="robert m hill" new_id="x97031807" sequence="31535"/>
  <individual timeframe="1920" timeframepre="0/0/1920:0/0/X" label="Stephens, Robert" labelpre="robert stephens" new_id="x97031807" sequence="41917"/>
  <individual timeframe="1923" timeframepre="0/0/1923:0/0/X" label="Richman, Robert J." labelpre="robert j richman" new_id="x97031807" sequence="45237"/>
  <individual timeframe="1905" timeframepre="0/0/1905:0/0/X" label="Smith, Robert" labelpre="robert smith" new_id="x97031807" sequence="46283"/>
  <individual timeframe="1932" timeframepre="0/0/1932:0/0/X" label="Smith, Robert H." labelpre="robert h smith" new_id="x97031807" sequence="46417"/>
  <individual timeframe="1942" timeframepre="0/0/1942:0/0/X" label="Smith, Robert V." labelpre="robert v smith" new_id="x97031807" sequence="46633"/>
  <individual timeframe="1927" timeframepre="0/0/1927:0/0/X" label="Knapp, Robert C." labelpre="robert c knapp" new_id="x97031807" sequence="47893"/>
  <individual timeframe="1895" timeframepre="0/0/1895:0/0/X" label="Graves, Robert" labelpre="robert graves" new_id="x97031807" sequence="48889"/>
  <individual timeframe="1889-1969" timeframepre="0/0/1889:0/0/1969" label="Viksten, Albert" labelpre="albert viksten" new_id="x97031807" sequence="6321"/>
  <individual timeframe="1931-" timeframepre="0/0/1931:0/0/X" label="Walter, Robert" labelpre="robert walter" new_id="x97031807" old_id="x97031807" sequence="3158"/>
  <individual timeframe="1938-" timeframepre="0/0/1938:0/0/X" label="Fitch, Robert" labelpre="robert fitch" new_id="x02068135" old_id="x02068135" sequence="9380"/>
  <individual timeframe="1940-" timeframepre="0/0/1940:0/0/X" label="May, Robert" labelpre="robert may" new_id="x90189782" old_id="x90189782" sequence="15500"/>
  <individual timeframe="1923-" timeframepre="0/0/1923:0/0/X" label="Hilton, Peter" labelpre="peter hilton" new_id="x90091748" old_id="x90091748" sequence="16439"/>
  <individual label="Solomon, Robert" labelpre="robert solomon" new_id="x90055734" old_id="x90055734" sequence="18691"/>
  <individual timeframe="1931" timeframepre="0/0/1931:0/0/X" label="Rutherford, Robert B." labelpre="robert b rutherford" new_id="x90818774" old_id="x90818774" sequence="19703"/>
  <individual timeframe="1933" timeframepre="0/0/1933:0/0/X" label="Giles, Robert H." labelpre="robert h giles" new_id="x90604686" old_id="x90604686" sequence="22114"/>
  <individual label="Kuenne, Robert E." labelpre="robert e kuenne" new_id="x90171582" old_id="x90171582" sequence="11037"/>
  <individual timeframe="1932-" timeframepre="0/0/1932:0/0/X" label="Chilton, John" labelpre="john chilton" new_id="x90073910" old_id="x90073910"
```

sequence="25080"/>
<individual label="Aliber, Robert, Z." labelpre="robert aliber" new_id="x90178479" old_id="x90178479" sequence="25875"/>
<individual timeframe="1943-" timeframepre="0/0/1943:0/0/X" label="Anderson, Robert M." labelpre="robert m anderson" new_id="x90777988" old_id="x90777988" sequence="26189"/>
<individual timeframe="1947-" timeframepre="0/0/1947:0/0/X" label="Hart, Roger" labelpre="roger hart" new_id="x90922670" old_id="x90922670" sequence="28526"/>
<individual timeframe="1944-" timeframepre="0/0/1944:0/0/X" label="Morris, Robert" labelpre="robert morris" new_id="x97041680" old_id="x97041680" sequence="30059"/>
<individual label="Grant, Robert M." labelpre="robert m grant" new_id="x90936473" old_id="x90936473" sequence="34562"/>
<individual label="Jensen, Robert" labelpre="robert jensen" new_id="x90127123" old_id="x90127123" sequence="38014"/>
<individual label="Galland, Robert B." labelpre="robert b galland" new_id="x90350078" old_id="x90350078" sequence="40675"/>
<individual label="Schware, Robert" labelpre="robert schware" new_id="x90130993" old_id="x90130993" sequence="41254"/>
<individual label="Barnard, Robert" labelpre="robert barnard" new_id="x90072687" old_id="x90072687" sequence="2772"/>
<individual timeframe="1942-" timeframepre="0/0/1942:0/0/X" label="Leach, Robert" labelpre="robert leach" new_id="x90872290" old_id="x90872290" sequence="42405"/>
<individual timeframe="1933" timeframepre="0/0/1933:0/0/X" label="Croken, Robert C." labelpre="robert c croken" new_id="x90771577" old_id="x90771577" sequence="45527"/>
<individual timeframe="1923-" timeframepre="0/0/1923:0/0/X" label="Hilton, Peter" labelpre="peter hilton" new_id="x90091748" old_id="x90091748" sequence="45756"/>
<individual timeframe="1930" timeframepre="0/0/1930:0/0/X" label="Sparkes, Robert S." labelpre="robert s sparkes" new_id="x90543995" old_id="x90543995" sequence="47109"/>
<individual timeframe="1929-" timeframepre="0/0/1929:0/0/X" label="White, Roger" labelpre="roger white" new_id="x97006430" old_id="x97006430" sequence="48483"/>
<individual label="Haas, Robert" labelpre="robert haas" new_id="x90363389" old_id="x90363389" sequence="22115"/>
<individual label="Cohen, Robert S." labelpre="robert s cohen" new_id="x90067220" old_id="x90067220" sequence="48531"/>
<individual timeframe="1896-1985" timeframepre="0/0/1896:0/0/1985" label="Sessions, Roger" labelpre="roger sessions" new_id="x90237242" old_id="x90237242" sequence="49659"/>
</merge>

10 Conclusion

The dynamic architecture of the system, made it possible to replace modules by changing a properties file. And the overall information flow between the system's modules provided a basis for the chosen preprocessors and measures to do their thing.

Based on the evaluation the label preprocessors was working properly and contributed to finding ambiguous entries in the content it was given (except from the one false match in the 90% evaluation, and several in the 50% evaluation). The time-frame preprocessor contributed, an order of magnitude, more than the label preprocessors did. This was because it standardised the original date field to a more comparing friendly format (day/month/year:day/month/year).

The chosen label measurer was able to eliminate false positive matches on a 100% accuracy setting, but it failed to do so, on one occasion, when the accuracy was 90%. The 50 % accuracy setting performed very poor, which was expected.

The specially developed time-frame measurer performed very well in cooperation with the time-frame preprocessor. The measurer was able to allow missing information, which made the it possible to decide whether two time-frames where within each others scope. On 50% accuracy the *Disambiguater* module had a implementation dilemma, which made the time-frame measurer appear less robust than it really is.

Providing a search interface (JDBCAdapter) to seek out possible matches gave an advantage. The advantage was that one could quickly eliminate most of the collection from being a possible match. The possible matches were confirmed or declined by comparing the time-frame of the creators, or the label of any publications associated with it.

Another feature that was very useful, as we saw in the *Evaluation and Discussion* section, was the *Log* module. The module made evaluation of the chosen measures much easier than browsing the database itself.

The paragraphs above shows that the label measurer (Levenshtein edit-distance) performed better than any previous disambiguater did on the BIBSYS content. But it also shows that it is not the appropriate measurer for approximate disambiguation of bibliographic information.

The most positive outcome of the thesis was the dynamic system architecture, time-frame preprocessor and time-frame measurer. Future work on iDup would probably utilize an *Adapter* module which combine canopy clusters with the current filtering model. And also use other more advanced edit-distance measures in conjunction with some n-dimensional space on both letters and on words/sub-names. And also a disambiguation of publication titles would be feasible.

11 References

[Besser, 2002] Howard Besser, *The Next Stage: Moving from Isolated Digital Collections to Interoperable Digital Libraries*, First Monday, nr. 6, vol. 7, First Monday (June 2002), http://www.firstmonday.org/issues/index.html

[BIBSYS-MARC, 2006] *BIBSYS-MARC - Bibliografisk format*, BIBSYS (March 2006), http://www.bibsys.no/out/wcm/connect/BIBSYS/handbok/marc/marc.htm

[Bilenko-Mooney, 2003] Mikhail Bilenko, Raymond J. Mooney, *Adaptive duplicate detection using learnable string similarity measures*, Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, p. 39 - 48, ACM Press (2003), ISBN:1-58113-737-0, http://doi.acm.org/10.1145/956750.956759

[Buckland, 1997] Michael Buckland, *What is a "document"?*, Journal of American Society for Information Science, nr. 9, vol. 48, p. 804 - 809 (1997)

[Cleveland, 1982] Harland Cleveland, *Information as Resource*, The Futurist, p. 34 - 39 (December 1982)

[Cohen-Kautz-McAllester, 2000] William W. Cohen, Henry Kautz, David McAllester, *Hardening soft information sources*, Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, p. 255 - 259, ACM Press (2000), ISBN:1-58113-233-6, http://doi.acm.org/10.1145/347090.347141

[Cohen-Richman, 2002] William W. Cohen, Jacob Richman, *Learning to match and cluster large high-dimensional data sets for data integration*, Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, p. 475 - 480, ACM Press (2002), ISBN:1-58113-567-X, http://doi.acm.org/10.1145/775047.775116

[Eliot, 1934 T.S Eliot, *The Rock*, Faber & Faber (1934)

[FRBR-BIBSYS, 2005] Trond Aalberg, Ole Husby, Frank Berg Haugen, Christian-

Emil Ore, *FRBR i bibliotekkataloger*, BIBSYS (2005), http://www.bibsys.no/wps/wcm/resources/file/eb00344e4910f7b/FRBR_i_Bibliotekkataloger.pdf

[FRBR-OCLC, 2006] *OCLC Research Activities and IFLA's Functional Requirement for Bibliographic Records*, OCLC (2006), http://www.oclc.org/research/projects/frbr/

[Harter, 1997] Stephen P. Harter, Scholarly *Communication and the Digital Library: Problems and Issues*, Journal of Digital Information, nr. 1, vol. 1 (1997), http://journals.tdl.org/jodi/article/view/jodi-3

[Hernández-Stolfo, 1995] Mauricio A. Hernández, Salvatore J. Stolfo, *The merge/purge problem for large databases*, Proceedings of the 1995 ACM SIGMOD international conference on Management of data, p. 127 - 138, ACM Press (May 1995), ISBN:0-89791-731-6, http://citeseer.ist.psu.edu/stolfo95mergepurge.h

[Hiemstra, 2007 Djoerd Hiemstra, *Information Retrieval Modelling*, chapter in: Information Retrieval: Searching in the 21st Century, editors: Ayse Göker, John Davies, John Wiley & Sons (2007)

[InterParty, 2003] *InterParty workshop presentations*, InterParty (2003), http://www.interparty.org/interparty/presentatio

[Levy-Marshall, 1995] David M. Levy, Catherine C. Marshall, *Going Digital: A Look at Assumptions Underlying Digital Libraries*, Communications of ACM nr. 4, vol. 38, p. 77 - 84 (1995)

[Lu, 1999] Guojun Lu, *Multimedia Database Management Systems*, Artech House, Inc. (1999), ISBN:0-89006-342-7

[MarcXchange, 2006] *ISO/DIS 25577 Information and documentation – MarcXchange*, (2006), http://www.bs.dk/marcxchange/ISO_DIS_25577__E_.pdf

[MAVIS2, 1999] Robert Tansley, Mark Dobie, Paul Lewis, Wendy Hall, *MAVIS 2: an architecture for content and concept based multimedia information exploration*, Proceedings of the seventh ACM international conference on Multimedia (Part 2), p 203, ACM press (1999), ISBN:1-58113-239-5, http://doi.acm.org/10.1145/319878.319941

[McAllister, 1981] A. Stratton McAllister, Caryl McAllister, *A design for an online bibliographic database: The DOBIS-LIBIS database*, Information Processing & management, nr. 1, vol. 17, p. 27 - 38, Elsevier Inc. (1981), http://dx.doi.org/10.1016/0306-4573(81)90039-X

[McCallum-Nigam-Ungar, 2000] Andrew McCallum, Kamal Nigam, Lyle H. Ungar, *Efficient clustering of high-dimensional data sets with application to reference matching*, Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, p. 169 – 178, ACM Press (2000), ISBN:1-58113-233-6, http://doi.acm.org/10.1145/347090.347123

[ModernIR, 1999] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, *Modern Information Retrieval*, ACM press (1999), ISBN:0-201-39829-X

[Monge-Elkan, 1997] Alvaro E. Monge, Charles P.Elkan, *An Efficient Domain-Independent Algorithm for Detecting Approximately Duplicate Database Records*, Proceedings SIGMOD Workshop Research Issues on Data Mining and Knowledge Discovery, p. 23 - 29 (May 1997), http://www.cs.ucsd.edu/users/elkan/approxdup.ps

[Robertson, 1977] Stephen E. Robertson, *The probability ranking principle in IR*, Journal of documentation, nr. 33 p. 294 - 304 (1977), http://www.soi.city.ac.uk/~ser/pubs.html

[Salton, 1971] Gerald Salton, *The SMART Retrieval System*, Prentice Hall, Inc (1971)

[Smith-Waterman, 1981] T. F. Smith, M. S. Waterman, *Identification of Common Molecular Subsequences*, Journal of Molecular Biology, vol 147, p.195 -197 (1981)

[Snyman-Rensburg, 2000] M. M. M. Snyman, M. Jansen van Rensburg, *Revolutionizing Name Authority Control*, Proceedings of the fifth ACM conference on Digital libraries, p. 185 - 194 ,ACM press (June 2000), ISBN:1-58113-231-X, http://doi.acm.org/10.1145/336597.336660

[VIAF, 2006] Rick Bennet,Christina Hengel-Dittrich,Edward T. O'Neill,Barbra B. Tillett, *Linking Die Deutche Bibliothek and Library of Congress Name Authority Files* (August 2006),

http://www.ifla.org/IV/ifla72/papers/123-Bennett

[Weber, 2002] Jutta Weber, *MALVINE, LEAF and Kalliope: Some co-operation models*, Digital access to Book Trade Archives, p. 49 - 68, Academic Press Leiden (2002)

[Winkler, 1999] W. E. Winkler, *The state of record linkage and current research problems*, Technical report, Statistical Research Division (1999), http://citeseer.ist.psu.edu/255199.html

[Yee, 2005] Martha M. Yee, *FRBRization: FRBRization: a Method for Turning Online Public Finding Lists into Online Public Catalogs*, Information Technology and Libraries, nr. 3, vol. 24, p. 77 - 95, Library & Information Technology Association (2005), http://repositories.cdlib.org/postprints/715/