Matching.

Lessons learned from BibMatch at INSPIRE / INVENIO1

INSPIRE relies heavily on matching, since we merge preprint and journal record.
In general, whenever various sources are harvested, matching is needed.

BibMatch offers many possibilities for customization,
but some necessary features are not possible.

Content:

- how to get reasonable results out of BibMatch (INVENIO1) the way it is
- how to improve performance with the matcher (INVENIO3)

Audience:

- Ordinary people
- Developer

Kirsten Sachs
IUGW, Garching
22.3.2017

HELMHOLTZ ASSOCIATION

DESY

How to use BibMatch.

```
bibmatch [options] -q query < new_records.xml
```
from STDIN to STDOUT using one query

```
bibmatch [options] -c queries -n -b outname -i new_records.xml
```
from file to files using set of queries

What happens:
- From search-query (based on new record and query) get candidates
- Validate candidates
- If no records found: generate fuzzy query and search again

```
useful options:
-o o    set operator to ||, default: &
--no-validation
--no-fuzzy
```
(see next slide)

```
invenio-local.conf:
CFG_BIBMATCH_FUZZY_WORDLIMITS
CFG_BIBMATCH_MATCH_VALIDATION_RULESETS
```
used for search

used for validation

DESY

Generate Search-Query using BibConvert.

```
bibmatch -q "[037__a::ADD(037__a:,)]" -o o < new_records.xml
```

Query: `[037__a::ADD(037__a:,)]`

Record:
```
037__ $$aDESY-15-129
037__ $$aNIKHEF-2015-024
037__ $$aarXiv:1601.05088
```

→ List generated by BibConvert:

```
[037__a:DESY-15-129, 037__a:NIKHEF-2015-024, 037__a:arXiv:1601.05088]
```

Search-query = operator.join(List)

`037__a:DESY-15-129` **or** `037__a:NIKHEF-2015-024` **or** `037__a:arXiv:1601.05088`

Several fields are connected by operator, which is defined **ONCE** per call, default = &, use `option -o o` to change this.

Search-Query authors.

Record: `100__ $$aSachs, Kirsten`

Query: `[100__a::ADD(100__a:,)]`
Squery: `100__a:Sachs, Kirsten`

This doesn't work!!

Query: `[100__a::LIMW(COMMA,R)::ADD(author:",*")]`
Squery: `author:”Sachs, *”`

You have to search for the full name of the new record or last name only, there is no way to use the initial!

Proper handling of author names happens **ONLY** in validation.

This will improve with elasticsearch in INVENIO3.

Search-Query titles.

Record: `245__ $$aLyth bound of inflation with a tilt`

Query: `[245__a::ADD(title:",")]`
Squery: `title:"The Lyth bound of inflation with a tilt"`

You think this works?

No it doesn't because you are looking for
`Lyth bound of inflation with a tilt`

Query:
```
[245__a::DOWN::EXP(-,1)::REP( with , )::REP( from , )::SUP(PUNCT, )
::SUP(NUM,)::MINL(4)::WORDS(3,R)::ADD(title:)::REP( , title:)]
```
Squery: `title:lyth title:bound title:inflation`

Btw: The automatically generated fuzzy search query would be
(number of words is configurable):
`245__a:Lyth or 245__a:bound or 245__a:inflation or 245__a:with or 245__a:tilt`

Not very useful!

Need a dedicated way to create a fuzzy search.

DESY

Title examples: journal / arXiv.

125 GeV Higgs signal at the LHC in the CP-violating MSSM
The 125 GeV Higgs signal at the LHC in the CP Violating MSSM

Systematizing semishortening conditions
Systematizing semi-shortening

Potential of the Large Observatory for X-ray Timing telescope for the search for dark matter
Potential of LOFT telescope for the search of dark matter

Majorana dark matter in warped extra dimensions
A Heavy "Neutralino" in Warped Extra dimensions

Trilinear self-couplings in an S(3) flavored Higgs model
S(3) flavoured Higgs model trilinear self-couplings

Top squark with mass close to the top quark
Stop on Top

Screening of scalar fields in Dirac-Born-Infeld theory
D-BIonic Screening of Scalar Field

Symmetry relations between angular observables in $B^0 \to K^* \mu^+\mu^-$ and the LHCb $P_5^\prime$ anomaly
A new relation between the zero of A_{FB} in $B^0 \to K^* \mu^+\mu^-$ and the anomaly in $P_5^\prime$

You might need a very fuzzy title search to find the article!

Shorter, hyphens, acronyms, latex/UTF8/ascii, … and completely different titles.
APS is notorious for changing the title.

DESY

List of Queries.

```
bibmatch [options] -c queries -n -b outfile -i new_records.xml
```

queries is a config file:

```
QRYSTR---@OR
QRYSTR---[037__a::ADD(037__a:,)]---
QRYSTR---[0247_a::ADD(0247_a:,)]---
QRYSTR---@AND
QRYSTR---[100__a::LIMW(COMMA,R)::ADD(author:",*")] [773__w::ADD(773__w:)]---
QRYSTR---[100__a::LIMW(COMMA,R)::ADD(author:",*")] [245__a::DOWN::EXP(-,1)::REP(
```

In a rough way I enabled switching the operator via the config file. (apologies)

You can define a series of queries.
It will step through until a matching record is found.

This way it is possible to define dedicated fuzzy queries:
First search for identifiers: RepNo, DOI
Then try author + CNUM (conference ID)
Last try author + title

No need for automatically generated fuzzy query
if predefined queries have a 'fuzziness' parameter.

DESY

How it should work.

2 step process: search + confirm

E.g. it is not possible to search for the abstract / full-text,
but it can be compared to confirm the match.

Search: high efficiency, low purity

Goal: reduce $\mathcal{O}(1{,}000{,}000)$ records to < $\mathcal{O}(100)$ records
the result must contain the matching record.
Get it clean enough, so you can afford fancy confirmation.

Confirmation: high purity

don't present too much noise to cataloger
if the result is noise only, the next fuzzy step is not triggered

Search components.

Working with BibConvert is a nightmare.
Find a good compromise between predefined algorithm and configurable values.
I would like to say something like (just examples - syntax is up to you):

term	Meaning
authors[2/3]	take max. 3 authors [A, B, C], search for combinations of 2 of them: (A and B) or (A and C) or (B and C)
title[4]	take max. 4 best words of title connected with and, if there are several titles connect with or: (A1 and B1 and C1 and D1) or (A2 and B2 and C2)
title[strict]	predefined strict algorithm

You can build dedicated algorithms or work with elastic search.
Need different levels of 'fuzziness'.

I want to build simple queries like:
(authors[10] or collaboration) and title[strict]

DESY

The Big Five.

The most common terms used for matching deserve predefined behavior.

	DOI	RepNo	PubNote	Title	Author
Connecting operator	||	||	||	||	&
Modification or elastic search	Strip http://dx.doi.org	none	Use journal name variants	Various possibilities	first-names → initials
other					Max. number
Fuzzy examples	none	strip \W	Only journal name	More possibilities	Combinations of subsets

Other terms might be used. E.g.

- CNUM – conference identifier
- Collaboration

Need a generic term with basic possibilities for customization.
Define operator for each term.

Fuzzy vs. Multi-Step Search.

If no record is found BibMatch generates automatically a fuzzy search.

Is there a generic algorithm to create a fuzzy search?
What is the meaningful fuzzy search for DOI, report-number, title, author?

BibMatch uses
`operator.join([N longest words in string])`
where operator and N can be configured
It's still searching for all authors and even for title this makes no sense.

Better give the user the tools to define a search tree and less restrictive searches:

1. DOI
2. RepNo
3. PubNote
4. title[strict] and author[5]
5. title[1/4] and author[1/3] and PubNote[JournalOnly]

Maybe even a possibility to adjust the confirmation.

Confirmation.

Can be very powerful but needs to be tuned carefully.

Text:
Similarity of titles, abstracts
In principle also references, full-text, ...

Authors:
Number of authors similar?
0 or 1 authors matches any (only collaboration or 1st author given)

Numbers:
Year, number of pages, … Can be very useful but don't ask for '=='

Type:
Journal article != Thesis

To be really useful you have to define how to compare the terms.
A generic '==' can only be a fall-back solution.

Strict search (DOI) might go with fuzzy confirmation (cross-check), fuzzy search needs strict confirmation.

Output.

BibMatch: matched (1 record found),
ambiguous (more than 1 record found),
fuzzy (result of fuzzy search),
new

What I want to know:
exact (1 record found, very high probability, can be merged automatically),
ambiguous (1 or more records found, to be confirmed by cataloger),
new
+ probability and query-of-discovery.

Again BibMatch has a feature (structured output) that
doesn't correspond to the needs of the user.

**Think what is most useful information,
not which information is easiest to extract.**

Summary.

What we have:

- Learn BibConvert if you want to work with BibMatch
- Carefully look at options and config variables,
  they might not correspond to what you expect
- Might need a user interface

What we want:

- Creating a search-query is easier in python
  → need predefined algorithms for the Big5 (RepNo, DOI, PubNote, author, title)
- Different levels of fuzziness for predefined algorithms
- Adjust confirmation by function call

To get good performance a generic algorithm is not enough,
even if it's smart and can be configured.
Including knowledge about the metadata is essential.

DESY