This document has the same content as `README.markdown`.

MerkOrCore

Version 0.8
Copyright 2012 Anna Björk Nikulásdóttir
Website: http://merkor.skerpa.com
Contact: anna.b.nik@gmx.de

MerkOrCore is an API for the access of the MerkOr, a semantic database for Icelandic.

License

The MerkOr project was funded by a grant from the Icelandic Research Fund (RANNÍS), grant nr. 090662011, Viable Language Technology beyond English - Icelandic as a test case.

About MerkOr

MerkOr is an automatically constructed semantic database for Icelandic. The basic elements of the database are:

- lexical item. Contains an id, a 'lemma' (=word string), sense number and a wordclass.
    - [id=109799, lemma=skúr_1, wordclass=noun]
- relation. A relation connects two lexical items with a relation type (see next). Each relation has a confidence score associated to it, the higher this score, the better / more representative the relation.
    - [id=893, from_item_id=52069, relation_id=7, to_item_id=34948, confidence_score=366.806]
- relation type. Specifies the type of relationship between two lexical items
    - [id=7, name=og, description=og]
- cluster. A cluster is an ordered list of lexical items belonging to the same semantic domains. Each item in a cluster has a score associated to it, indicating how well the item fits the corresponding cluster. Less than 10,000 items belong to a cluster.

The MerkOrCore API and command line interface can be used to query this data:

- Does a word belong to more than one lexical item?
- Which relations exist for a certain word?
- What are the relations with the highest confidence score for a certain word?

- What are the relations with the highest confidence score for a certain relation type?
- To which cluster(s) does a word belong to?
- Are there clusters representing some certain semantic domain (like ÍÞRÓTTIR*)?
- Which lexical items are connected to a certain domain?
- etc. See instructions below!

Getting started

MerkOrCore is developed and tested under Mac OS X only – please report any problems with other platforms.

Redis

The MerkOr data is stored in Redis format (redis version 2.4.5). Redis is available at http://redis.io (installation instructions under http://redis.io/download).
The Redis data is included in this package as \`dump.rdb\`.
After you have installed Redis and loaded the MerkOr data, you can try it out directly in Redis command line interface, (in the Redis directory start \`src/redis-cli\`), for example:

```
redis 127.0.0.1:6379> smembers merkor_is_lemma_lampi
  1) "merkor_is_id_45966"
  redis 127.0.0.1:6379> hgetall merkor_is_id_45966
  1) "lemma"
  2) "lampi"
  3) "wordclass"
  4) "noun"
```

This assures you that the MerkOr data is loaded but you don't have to study redis-cli to use the MerkOrCore API. For those interested in inspecting the data directly, for example with redis-cli, the structure of the Redis data is shown in the file \`merkor_redis_structure.txt\`.

MerkOrCore command line interface

In the initial project state a file MerkOrCore.jar is included in the release folder. I recommend, however, to rebuild the project by typing `ant` in the folder you saved the project to (preferably MerkOrCore).

Before running MerkOrCore, either as a command line interface or as an API, **make sure the Redis server is running!**.

In the directory of MerkOrCore.jar type:

```
java -jar MerkOrCore.jar -help
```

to see the parameters available.

The default configuration for Redis implemented in MerkOrCore is "localhost" and port 6379.
If you have another configuration you have to use the `-host` and `-port` parameters to specify it.

Possible combinations:
Get all items for a lemma

Some words (=lemma) belong to more than one lexical item, e.g. because they have different wordclasses or different gender. To see if this is true for a lemma in MerkOr, type:

```
java -jar MerkOrCore.jar -items <lemma>
```

Typing *skúr* as lemma, which exists both as a noun with gender mask. and fem., should give the result:

```
lexical item: [id=109799, lemma=skúr_1, wordclass=noun]
 lexical item: [id=112793, lemma=skúr_1, wordclass=noun]
```

Get relations for a lemma
To get all relations containing the given lemma, type:

```
java -jar MerkOrCore.jar -relations <lemma>
```

All relations have a *confidence score*. The higher this score, the likelier the relation.
You can add a number parameter and thus only get the top `n` relations according to the confidence score:

```
java -jar MerkOrCore.jar -relations <lemma> -n <integer>
```

Requesting relations for a lemma always returns relations for all items the lemma belongs to.
Thus requesting for example the top 10 relations for *skúr* returns two lists of ten relations for each item *skúr* belongs to.

You can get all relations of a certain type, having a certain lemma as its left or right element:

```
java -jar MerkOrCore.jar -rel_from <lemma> -rel_type <relation_type>
```

or
```
   java -jar MerkOrCore.jar -rel_to <lemma> -rel_type <relation_type>
```

Try for example the lemma 'heitur' as -rel_from and the relation type 'lýsir' as relation type.
All relation types are listed in `merkor_relationTypes.csv`. All names in this file can be used as arguments for `-rel_type`, just be sure to use `"` or `'` around names containing spaces.

To see if two words are related, use `-rel_from` and `-rel_to`:

```
java -jar MerkOrCore.jar -rel_from <lemma1> -rel_to <lemma2>
```

Get relations by type
All relation types are listed in `merkor_relationTypes.csv`. All names in this file can be used as arguments for `-rel_type`, just be sure to use " or ' around names containing spaces.
Get the `n` relations of a certain type with the highest confidence score:

```
java -jar MerkOrCore.jar -rel_type <relation type> -n <integer>
```

If you don't specify the number of relation, a maximum of 100 relations will be displayed (some relation types do not have that many relations, in these cases all relations are shown).

Clusters and domains

A cluster is a automatically constructed set of semantically related words. Most of them have been assigned a domain name, like 'ÍÞRÓTTIR' (='SPORTS') or 'FJÁRMÁL' (='FINANCE'). These names are not unique for a cluster (they have a unique id), so some domain names may have more than one cluster associated to it.

Get all cluster names:

```
java -jar MerkOrCore.jar -clusters
```

Names in upper case are domain names assigned by the author, names in lower case represent clusters that do not have a domain name assigned to them. A lower case name is the central word of its cluster and thus gives an idea about the domain of the cluster.

Get a cluster by its id:

```
java -jar MerkOrCore.jar -cluster_id <cluster_id>
```

Get clusters by name / regular expression ('?' and '*' are allowed wildcards):

```
java -jar MerkOrCore.jar -clusters_matching <regex>
```

Get clusters containing a lemma:

```
java -jar MerkOrCore.jar -clusters_having <lemma>
```

Get domains a lemma belongs to:

```
java -jar MerkOrCore.jar -domains_having <lemma>
```

Compare for example the output of the two last commands using the lemma 'skip'.

Get all items belonging to a certain domain (try this one for example for the domain 'skip'):

```
java -jar MerkOrCore.jar -items_for_domain <domain>
```

Get all items belonging to a certain cluster:

```
java -jar MerkOrCore.jar -items_for_cluster <cluster id>
```

For this command you need to know the id of the cluster you want to inspect. Call -clusters_matching <regex> to get ids of clusters matching the domain you're interested in. You can also use any integer between 1 and 305 if you are not interested in a particular cluster.

The output shows the items, cluster name and id, and values. Value is between 0.25 and 1.0, the higher the better the item fits into the cluster.