This page last changed on Jan 31, 2007 by mikebe.

Running the By-Example crawl in Heritrix

This section provides an actual example of crawl job on topic of jazz-music, which includes all the algorithm steps: pre-processing, clustering and classification. The example will provide instructions on how to run each step and how to analyze the steps output.

Pre-requisites

- Java 1.5 JRE/JDK installation
- Heritrix (ver. 1.8, 1.10.1 or 1.10.2) distribution, including *by-example* plugin
- Knowledge of running crawl-jobs in Heritrix is required

Pre-processing step

The purpose of this step is to index the crawled pages. This indexing will be used in clustering step

Setup

- A new crawl job is created called *Jazz-music-PreProcessingJob*. Seeds include the following links:

```
http://www.harlem.org/people/name.html
http://www.jass.com/
http://www.jazzhouse.org/library/index.php3?sel=Book+Excerpts+and+Reviews
http://www.smallsjazz.com/
```

- Job modules:
  - Job is based on *default* profile
  - *org.archive.crawler.byexample.processors.TermsIndexingProcessor* is added to *Extractors* list
  - In *SubModules*:
    - *OnDomainDecideRule* is added to the DecidingScope, to limit crawl to the seed domains only
    - *Relevance Filter* is added to *TermsIndexingProcessor* submodules list (see **Figure 1a**). All the pages that will pass this filter will be deemed "relevant" by the processor. In this example *OnDomainDecideRule* is selected as relevance indication rule, however any other rule can be selected as well.
  - In *Settings*, a name for a crawl-by-example job ID can be set. By default, crawl-by-example job ID is set to be *byexamplejob-/Unique TimeStamp/*.

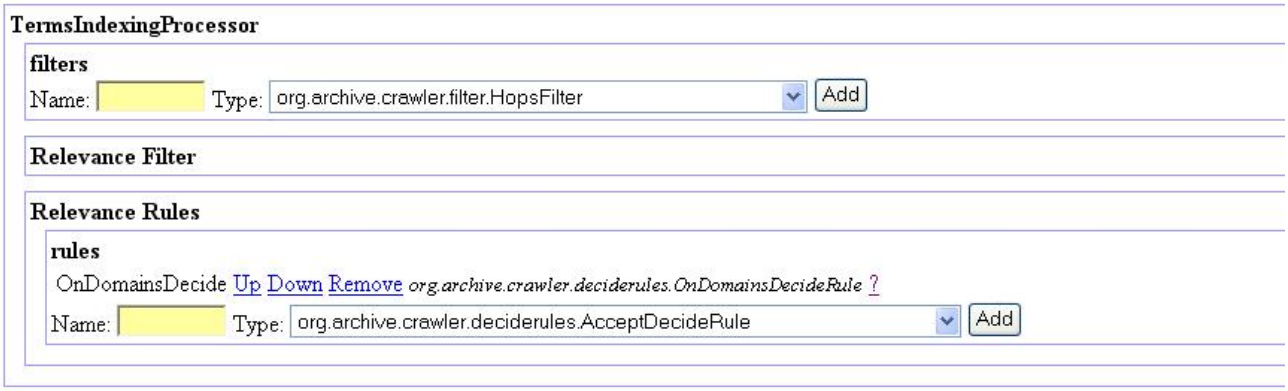

Figure 1a: Setting the filters for TermsIndexingProcessor

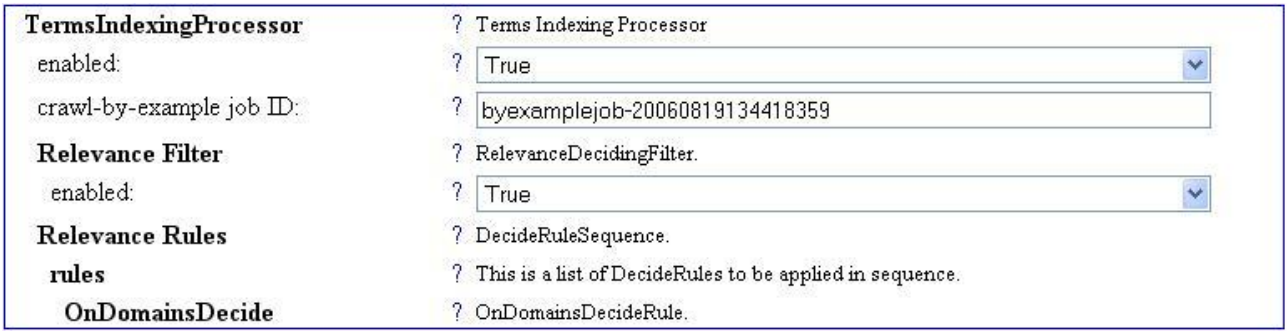

Figure 1b: Setting crawl-by-example job ID for TermsIndexingProcessor

- After all the modules are set, job can be started

Results analysis

After *Jazz-music-PreProcessingJob* is completed, following output is created under *HERITRIX-HOME/byexample/jobs/_byexamplejob-JobID* folder:

- *preprocess* folder. Folder contains the following files:
  - *documentListing.txt* - Crawled documents list, incl. id, url and relevance indicator for each crawled page
  - *termIndex.txt* - An inverted file index. See this wikipedia article for more details on the structure of inverted index.
- *preprocess-results.xml* - This is a summary of pre-process crawl results, incl. number of indexed terms, processed documents and location of output files.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <preProcess-output>
    <termsIndexFN>byexample\jobs\byexamplejob-20060818111939921\preprocess\termIndex.txt</termsIndexFN>
    <documentListingFN>byexample\jobs\byexamplejob-20060818111939921
      \preprocess\documentListing.txt</documentListingFN>
    <preprocessDocsNo>708</preprocessDocsNo>
    <preprocessTermsNo>13601</preprocessTermsNo>
  </preProcess-output>
```

Figure 2: Structure of preprocess-results.xml

Clustering step

Once the pre-processing crawl is completed, clustering step can be initiated. The purpose of this step is creating a clustering structure for the pre-processed pages.

Setup

- Prior to executing the step, clustering parameters should be configured. These parameters reside in *HERITRIX_HOME/byexample/conf/byexample.properties* file. Detailed explanation on each of the parameters is available in the properties file itself and in javadoc for class *org.archive.crawler.byexample.constants.AlgorithmConstants*
  Note: changing parameters values can significantly increase clustering run-time, especially for large document collections. See more detailed explanations inside the properties file and javadoc.
  Below is the parameters setup used in the jazz example:

```
min_global_support = 15
max_depth = 3
max_1_frequent_terms = 150
min_cluster_support = 70
min_size_to_prune = 3
unclassified_label = $UNCLASSIFIED$
top_classifications =3
top_relevant = 10
```

Figure 3: Clustering parameters

- The step itself is performed outside of Heritrix web UI. It can be invoked from a command line by executing the main class *org.archive.crawler.byexample.algorithms.clustering.ClusteringRunner*. Command line should look something like this:

```
HERITRIX_HOME> java -cp #BIN#
org.archive.crawler.byexample.algorithms.clustering.ClusteringRunner #JOB-ID#
```

where:

- *#BIN#*- jar containing the heritrix classes and Berkeley DB jar. E.g., if command is run under *HERITRIX_HOME/bin*, then #BIN# is ..\lib\byexample.jar;..\lib\je-3.0.12.jar\
- *#JOB-ID#* - JobID of the job created in the previous step (by default JobID would look like *byexamplejob-/Unique TimeStamp/*)
- If JobID is correct and all binary and configuration files are found in the classpath *ClusteringRunner* should be executed and write the sequence of its actions on the screen, including run-time for each action:

```
Aug 18, 2006 4:22:33 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: BUILDING FREQUENT TERMS INVERTED INDEX completed in 0 minutes, 0 seconds

Aug 18, 2006 4:23:56 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: CREATING 1535 FREQUENT ITEM SETS completed in 1 minutes, 0 seconds

Aug 18, 2006 4:24:01 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: CREATING INITIAL CLUSTERS completed in 0 minutes, 4 seconds

Aug 18, 2006 4:24:02 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: CALCULATING DOCUMENTS SUPPORT SCORES completed in 0 minutes, 1 seconds

Aug 18, 2006 4:24:04 PM org.archive.crawler.byexample.utils.TimerUtils reportPartialAction
INFO: Completed so far 9% of the action: SUPPORT CALCULATION
Aug 18, 2006 4:24:05 PM org.archive.crawler.byexample.utils.TimerUtils reportPartialAction
```

```
INFO: Completed so far 19% of the action: SUPPORT CALCULATION
....
....
....
INFO: Completed so far 99% of the action: SUPPORT CALCULATION
Aug 18, 2006 4:24:17 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: CALCULATING INITIAL CLUSTERS SUPPORT SCORES completed in 0 minutes, 14 seconds

Aug 18, 2006 4:24:31 PM org.archive.crawler.byexample.utils.TimerUtils reportPartialAction
INFO: Completed so far 9% of the action: DISJOINING CLUSTERS
Aug 18, 2006 4:24:45 PM org.archive.crawler.byexample.utils.TimerUtils reportPartialAction
INFO: Completed so far 19% of the action: DISJOINING CLUSTERS
....
....
....
Aug 18, 2006 4:27:18 PM org.archive.crawler.byexample.utils.TimerUtils reportPartialAction
INFO: Completed so far 99% of the action: DISJOINING CLUSTERS
Aug 18, 2006 4:27:19 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: DISJOINING CLUSTERS completed in 3 minutes, 1 seconds

Aug 18, 2006 4:27:19 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: MERGING SMALL CLUSTERS completed in 0 minutes, 0 seconds

Aug 18, 2006 4:27:20 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: PRUNING EMPTY CLUSTERS completed in 0 minutes, 0 seconds

Aug 18, 2006 4:27:20 PM org.archive.crawler.byexample.utils.TimerUtils reportActionTimer
INFO: Action: CREATING CLUSTERING OUTPUT XML FILE completed in 0 minutes, 0 seconds
```

Figure 4: Clustering step actions

As can be seen, clustering step took about 4 minutes, given the parameters in **figure 3**.
***Note**: Just to exemplify the influence of clustering parameters on algorithm run-time, it can be reported that if **max_1_frequent_terms parameter** value is set to **50** (instead of **150**) execution is completed in under 20 seconds, although, of course, the resulting classification is expected to be less accurate.*

Results Analysis

After clustering is completed, following output is created under *HERITRIX-HOME/byexample/jobs/JobID* folder:

- *clustering* folder. Folder contains the following files:
  - *clusteringDocs.txt* - Contains mapping of document ids to created clusters. In conjunction with *documentListing.txt* from previous step, user can determine to which cluster specific url was mapped.
  - *clusteringTermSupport.txt* - Contains mapping of most frequent terms to created clusters. Each term is presented by *term::score* pair. Score is percentage of documents in the cluster containing the term. These scores are used in the next step to classify the crawled pages.
- *clustering-results.xml* - This is a summary of clustering results, incl. list of all created clusters and location of output files. Each cluster is presented by label (term(s) that appear(s) in all cluster documents), number of associated documents, estimated cluster relevance to crawl subject (jazz) and list of most frequent terms.

```
- <cluster>
    <clusterLabel>{band;blue;radio;}</clusterLabel>
    <clusterDocsNo>40</clusterDocsNo>
    <clusterRelevance>1.0</clusterRelevance>

      <associatedTerms>record;root;morgan;band;blue;song;orlean;tom;art;com;home;jazz;feedback;link;pictur;musician;
  </cluster>
```

Figure 5: Example of cluster representation from clustering-results.xml

Classification step

This step is another, larger crawl of the topic, which will classify the crawled pages according to clusters found in the previous step

Setup

- Prior to executing the step, classificaton parameters should be configured. These parameters reside in *HERITRIX_HOME/byexample/conf/byexample.properties* file, similarly to clustering parameters. Currently only two parammeters are used in classification step: *top-classifications* and *top-relevant*(see **figure 3**). *top-classification parameter defines the maximum number of classifications assigned to each crawled page. _top-relevant* defines the number of documents appearing in most/least relevant listings.

- A new crawl job is created called *Jazz-music-ClassificationJob*. Seeds include the following links:

```
http://www.42explore.com/jazz2.htm
http://www.jazzscript.co.uk/lifeline.htm
http://www.swingmusic.net/
```

- Job modules:
  - Crawl Scope is set to *BroadScope* with *max-link-hops,max-trans-hops* set to *1*.
  - *org.archive.crawler.byexample.processors.ClassifierProcessor* is added to *Extractors* list
  - In *Settings* following parameters are set:
    - *based-on* setting for *ClassifierProcessor* is set to *#JOB-ID#*, where *#JOB-ID#* is identical to JobId used in the previous steps (see **figure 6**). This means that *ClassifierProcessor* will use clustering results of this job.
    - *crawl-by-example job ID*. By default, it is set to be *byexamplejob-/Unique TimeStamp/*.

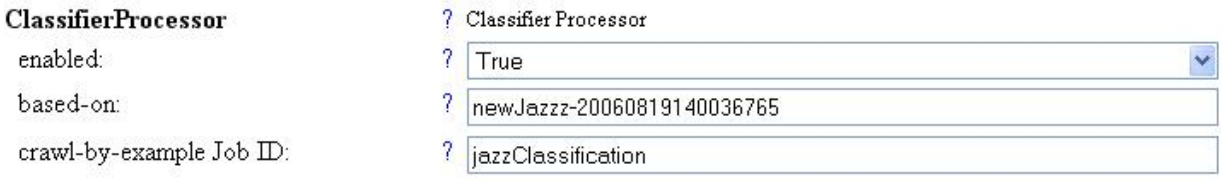

Figure 6: Setting the ClassifierProcessor

Results Analysis

After classification crawl is completed, following output is created under *HERITRIX-HOME/byexample/jobs/_byexamplejob-ClusteringJobID* folder:

- *classification* folder. Folder contains the following files:
  - classifiedDocs.txt - listing of all classified pages, incl. the following details about each page:
    - Page url
    - Page classifications. Each classification is assigned a score - the higher the score, the higher is the probability of assigning the page to the cluster.
    - Page relevance score
  - *unclassifiedDocs* - listing of all pages that couldn't be classified during crawl.

- *(most/least)RelevantList* - listing of pages that got highest/lowest relevance score during the crawl.

Some classification examples

During the classification step, around 2000 crawled pages were classified.

Some interesting examples for pages that were found to be most relevant on jazz topics:

```
...
http://members.aol.com/Jlackritz/jazz/~{johnni;louisiana;}::0.33041124939474054;{basin;}::0.3317416495364058
http://shs.starkville.k12.ms.us/mswm/MSWritersAndMusicians/music.html~{tuesdai;}::0.32735922266489986;{cd_ti
...
```

Some interesting examples for pages that were found to be least relevant on jazz topics or could not be classified at all

```
...
http://www.ampiramedia.com/~{$UNCLASSIFIED$;}::1.0;~0.0
http://www.real.com/~{$UNCLASSIFIED$;}::1.0;~0.0
http://www.redlightbydf.com/Bolden.html~{own;}::0.29587983365707615;{book;}::0.32273487836932285;{galleri;qu
http://www.historychannel.com/blackhistory/
...
```