Recuperación de Información con Terrier

Taller Libre I

Contenido

- Introducción
- Componentes
- Instalación
- Configuración / Preparación
- Ejecución

Introducción

Experimentación IR

Motores de búsqueda disponibles

No-Académicos

- Lucene (Apache)
- Minion (Oracle)
- Xapian (Cambridge)
- Sphinx (Sphinx Inc.)

Académicos

- Terrier (Glasgow)
- Lemur/Indri (CMU/UMass)
- Zettair

Introducción

"Terrier is a highly flexible, efficient, and effective open source search engine, readily deployable on large-scale collections of documents"

Eficiente

Efectivo

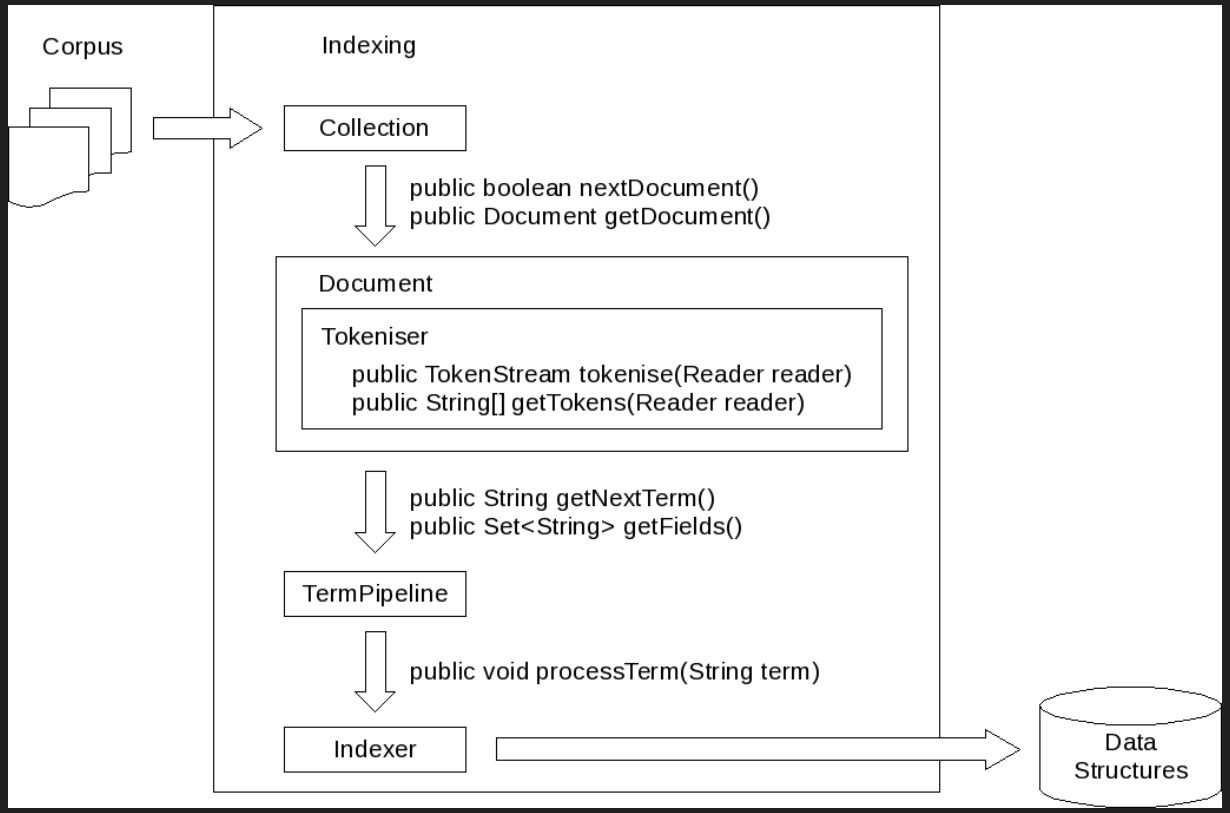
Flexible

Extensible

Multi-lenguaje

Interactivo

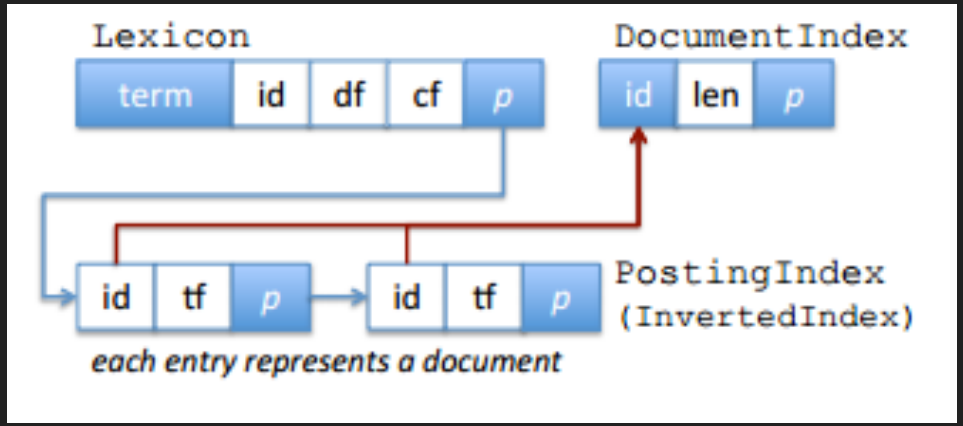
Componentes - Indexación



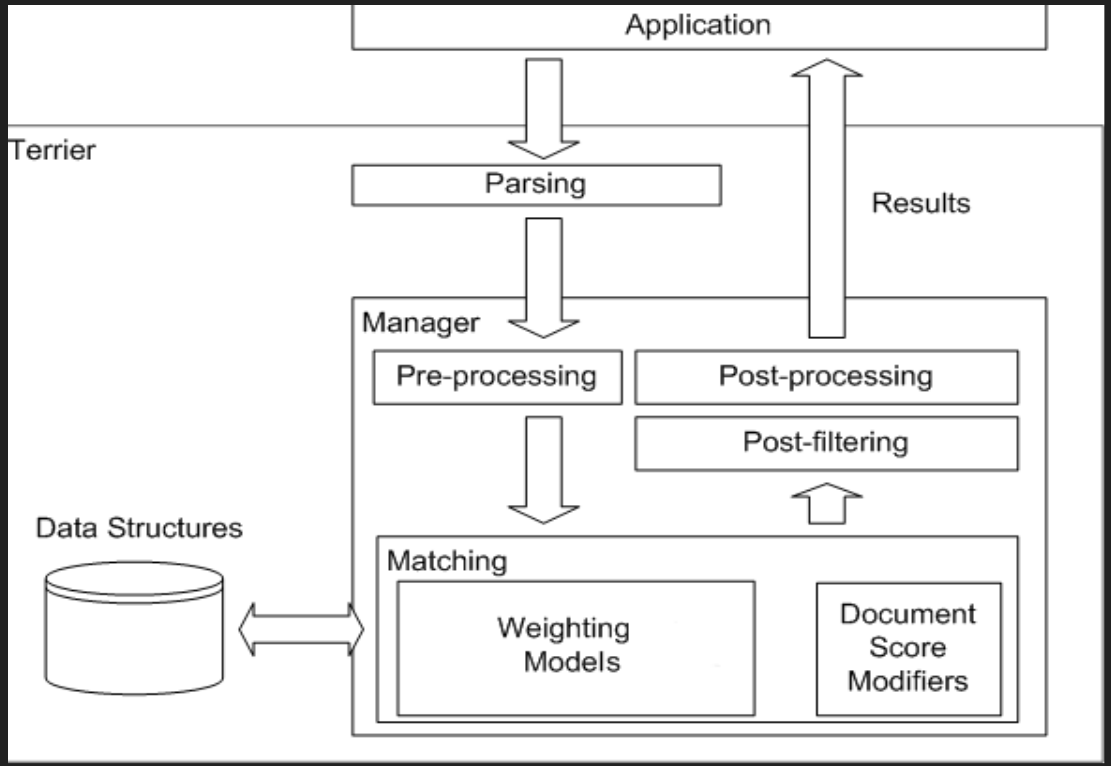
Componentes - Indexación

Estructuras generadas

- Lexicon
- Inverted index
- Document index (id doc, long.)
- Direct index (términos de cada doc y frecuencia)



Componentes - Recuperación



Instalación

- Descargar desde terrier.org (v4.0)
- Requerimientos
  - Java JDK 1.6+
- Ya compilado! Listo para usar
- Estructura de directorios

/bin - scripts para ejecutar Terrier

/etc - archivos de configuración

/lib - librerías java requeridas (.jar)

/share - archivos de utilidad (ej: stopwords)

/src - código fuente

/var - índices y resultados

Configuración / Preparación

- Configuración por defecto (/etc/terrier.properties)
- Soporte para configurar propiedades a través de la línea de comandos

- Documentación de todas las propiedades configurables terrier.org/docs/v4.0/properties.html

```
#default controls for query expansion
querying.postprocesses.order=QueryExpansion
querying.postprocesses.controls=qe:QueryExpansion
#default controls for the web-based interface. SimpleDecorate
#is the simplest metadata decorator. For more control, see Decorate.
querying.postfilters.order=SimpleDecorate,SiteFilter,Scope
querying.postfilters.controls=decorate:SimpleDecorate,site:SiteFilter,scope:Scope

#default and allowed controls
querying.default.controls=
querying.allowed.controls=scope,qe,qemodel,start,end,site,scope

#document tags specification
#for processing the contents of
#the documents, ignoring DOCHDR
TrecDocTags.doctag=DOC
TrecDocTags.idtag=DOCNO
TrecDocTags.skip=DOCHDR
#set to true if the tags can be of various case
TrecDocTags.casesensitive=false

#query tags specification
TrecQueryTags.doctag=TOP
TrecQueryTags.idtag=NUM
TrecQueryTags.process=TOP,NUM,TITLE
TrecQueryTags.skip=DESC,NARR

#stop-words file
stopwords.filename=stopword-list.txt

#the processing stages a term goes through
termpipelines=Stopwords,PorterStemmer
```

Configuración / Preparación

- Formatos de corpus
  - Archivos simples (SimpleFileCollection)
  - XML (SimpleXMLCollection)
  - Tweets (TwitterJSONCollection)
  - PDF, Excel, Word, …
  - TREC (TRECCollection)

```
<DOC>
<DOCNO> 1 </DOCNO>
Contenido del documento uno
</DOC>
<DOC>
<DOC>
<DOCNO> 2 </DOCNO>
Contenido del documento dos
</DOC>
<DOC>
...
```

```
#document tags specification
#for processing the contents of
#the documents, ignoring DOCHDR
TrecDocTags.doctag=DOC
TrecDocTags.idtag=DOCNO
TrecDocTags.skip=DOCHDR
#set to true if the tags can be of various case
TrecDocTags.casesensitive=false
```

Configuración / Preparación

- Formato de Query's (Topics)

```
<TOP>
<NUM>1<NUM>
<TITLE>house dog
<DESC>description
<NARR>narrative
</TOP>
<TOP>
<NUM>2<NUM>
<TITLE>chair
<DESC>description
<NARR>narrative
</TOP>
...
```

```
#query tags specification
TrecQueryTags.doctag=TOP
TrecQueryTags.idtag=NUM
TrecQueryTags.process=TOP,NUM,TITLE
TrecQueryTags.skip=DESC,NARR
```

- Lenguaje Terrier

  term1 term2 (term1 o term2)
  +term1 +term2 (term1 y term2)
  +term1 -term2 (term1 y no term2)
  "term1 term2" (term1 y term2 como frase)

Configuración / Preparación

- Modelos de recuperación
  - TF-IDF
  - BM25 (modelo probabilístico)
  - Hiemstra_LM (modelos de lenguaje)
  - … (lista completa en terrier.org/docs/v4.0/configure_retrieval.html)

```
#configure retrieval model
trec.model=BM25
```

- Juicios de relevancia
  - Archivo de 4 columnas, en el que se debe especificar id query, id doc y relevancia.

```
1   0   1   1
1   0   2   1
1   0   3   1
1   0   4   0
2   0   1   0
2   0   2   0
2   0   3   1
2   0   4   1
3   0   1   0
3   0   2   1
3   0   3   0
3   0   4   1
.
.
.
```

id query

id doc

relevancia (0-1)

Ejecución

1) Setup

>> ./trec_setup.sh ruta/absoluta/coleccion

En /etc se crea un archivo collection.spec con los archivos que se van a indexar

2) Indexación

>> ./trec_terrier.sh -i
>> ./trec_terrier.sh -i -Dtrec.collection.class=SimpleFileCollection

Se crean las estructuras de datos dentro de /var/index. Se pueden ver estadísticas de las mismas ejecutando:

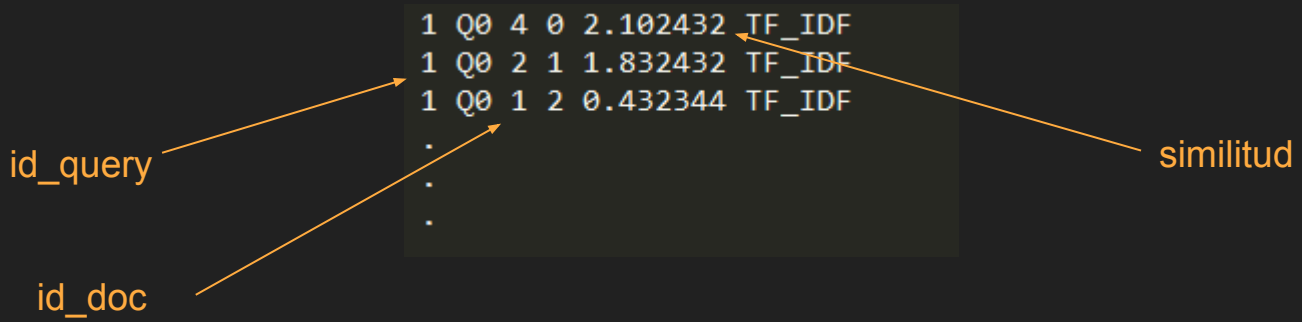
>> /bin/trec_terrier.sh --printstats
>> /bin/trec_terrier.sh --printdocid
>> /bin/trec_terrier.sh --printlexicon
>> /bin/trec_terrier.sh --printinverted

Ejecución

3) Recuperación

```
>>  ./trec_terrier.sh -r
>>  ./trec_terrier.sh -r  -Dtrec.model=BM25 -Dtrec.topics=/ruta/absoluta/topics
```

Dentro de /var/results se crea un archivo .res con los resultados de la recuperación



Ejecución

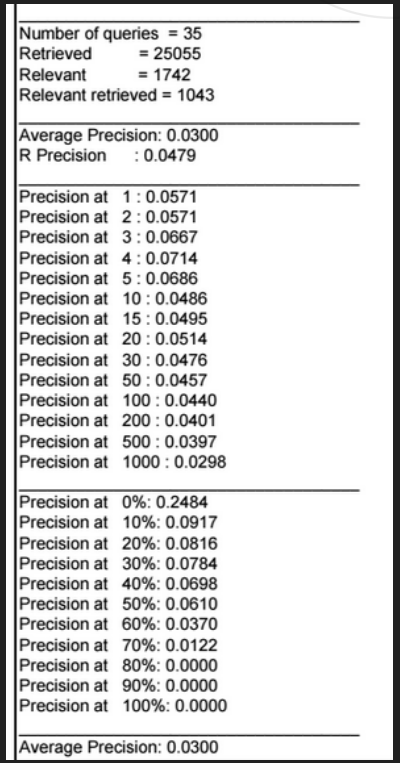
4) Evaluación

>> ./trec_terrier.sh -e

>> ./trec_terrier.sh -e -Dtrec.qrels=/ruta/absoluta/qrels

Dentro de /var/results se genera un archivo .eval con

los resultados de la evaluación



```
Number of queries  = 35
Retrieved          = 25055
Relevant           = 1742
Relevant retrieved = 1043

Average Precision: 0.0300
R Precision      : 0.0479

Precision at  1 : 0.0571
Precision at  2 : 0.0571
Precision at  3 : 0.0667
Precision at  4 : 0.0714
Precision at  5 : 0.0686
Precision at  10 : 0.0486
Precision at  15 : 0.0495
Precision at  20 : 0.0514
Precision at  30 : 0.0476
Precision at  50 : 0.0457
Precision at  100 : 0.0440
Precision at  200 : 0.0401
Precision at  500 : 0.0397
Precision at  1000 : 0.0298

Precision at  0%: 0.2484
Precision at  10%: 0.0917
Precision at  20%: 0.0816
Precision at  30%: 0.0784
Precision at  40%: 0.0698
Precision at  50%: 0.0610
Precision at  60%: 0.0370
Precision at  70%: 0.0122
Precision at  80%: 0.0000
Precision at  90%: 0.0000
Precision at  100%: 0.0000

Average Precision: 0.0300
```

FIN