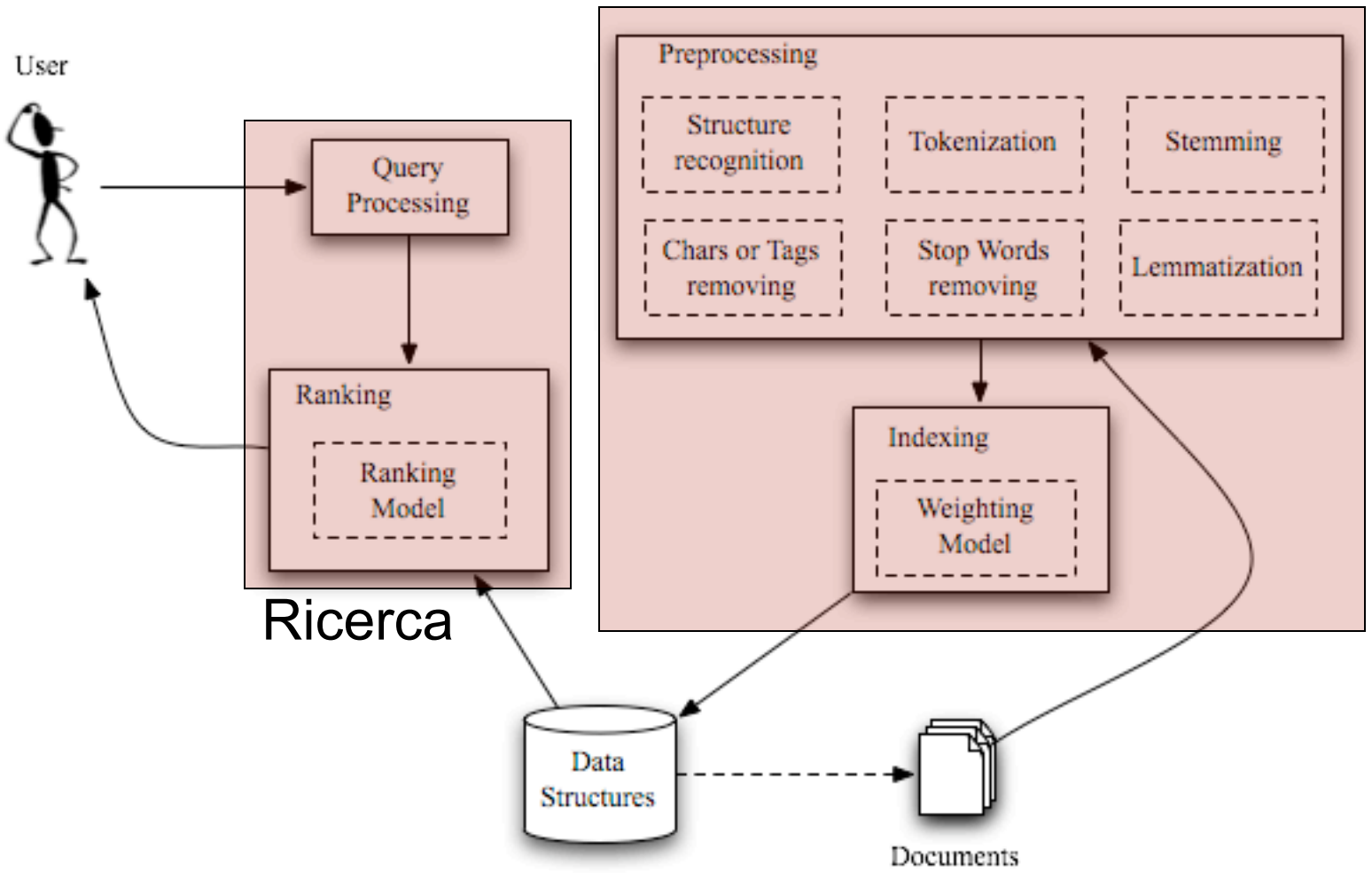
Introduzione a Lucene

Giuseppe Castellucci
castellucci@ing.uniroma2.it
Original version by Diego De Cao , Roberto Basili
Web Mining and
Information Retrieval
a.a. 2015/2016

Outline

- Uno sguardo a Lucene
- Descrizione delle principali caratteristiche
- Realizzazione di un semplice motore di IR
- Luke: un interfaccia grafica per Lucene
- Uno sguardo a *Lemur*

Architettura generica di un sistema di IR



Lucene

- Lucene è una libreria che permette di avere in Java un motore di ricerca full text.
- E' un progetto open-source della *Apache Software Foundation*
- Altamente scalabile e personalizzabile
- Esistono molte implementazioni in differenti linguaggi che condividono l'accesso agli stessi indici (C, C++, .NET, Python, Perl, Ruby, Php, …).
- http://lucene.apache.org/core/3_6_1/api/all/index.html

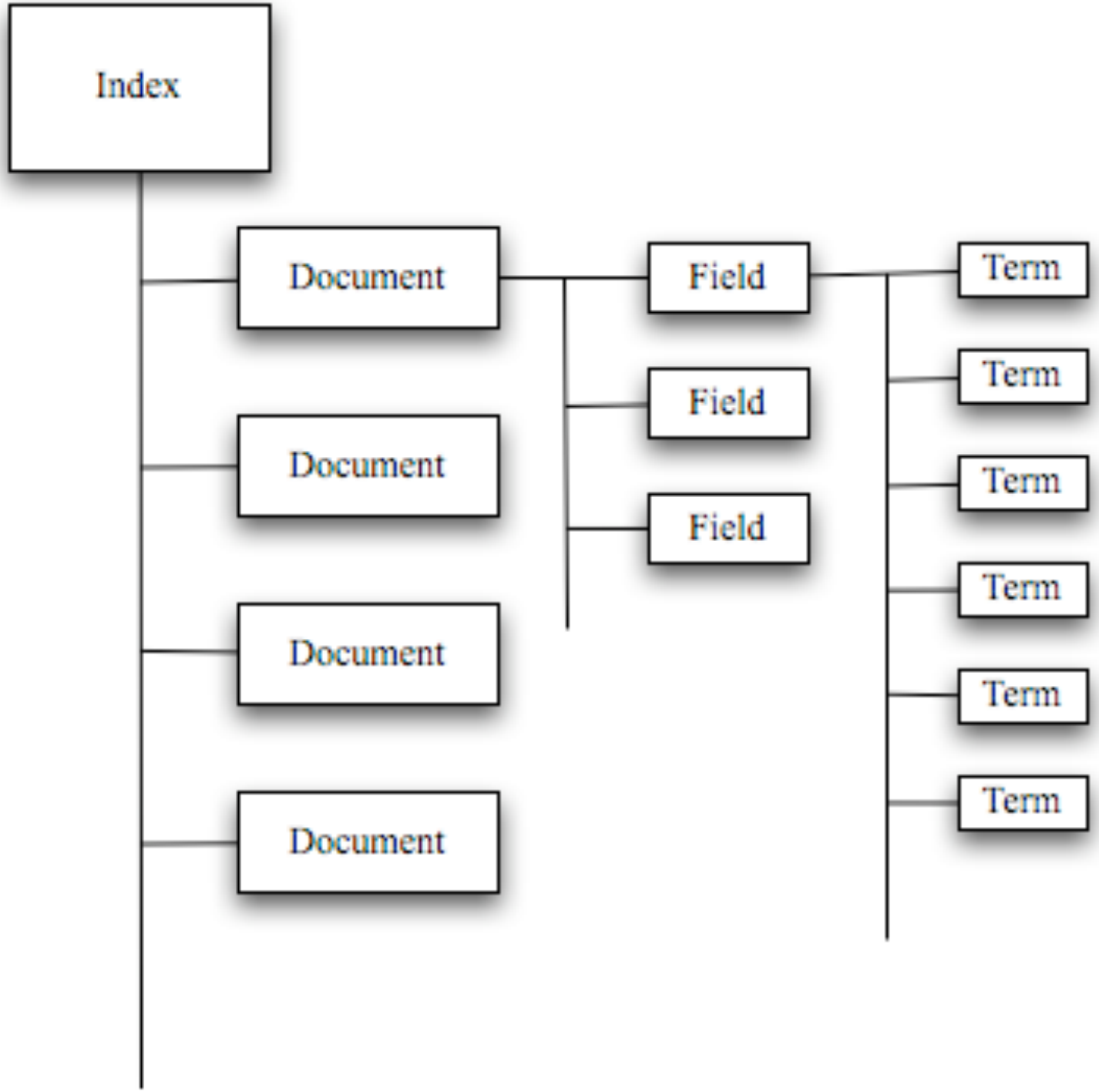
Lucene - caratteristiche principali

- Indicizzazione strutturata per campi
- Diversi strumenti di preprocessing per diverse lingue
- Differenti tipologie di Query:
  - phrase queries, wildcard queries, proximity queries, range queries, etc…
- Ricerca per campi
  - Singoli
  - Multipli (i risultati vengono uniti)

Concetti fondamentali: rappresentazione logica dell'indice

In Lucene l'indice permette la rappresentazione di un indice in modo strutturato.

- L'***indice*** contiene una sequenza di ***documenti***.
- Un ***documento*** e' composto da una sequenza di ***campi***
- Un ***campo*** e' una sequenza ***termini***
- I ***campi*** sono indipendenti. La stessa stringa che occorre in due ***campi*** differenti rappresenta due ***termini*** diversi


Index
Document
Document
Document
Document
Field
Field
Field
Term
Term
Term
Term
Term
Term

Struttura interna dell'indice

- **Field names**: Contiene i nomi dei campi di un documento.
- **Stored Field values**: per ciascun documento indicizzato, all'interno di questo campo viene memorizzata una coppia attributo-valore, dove gli attributi corrispondono ai nomi dei campi.
- **Term dictionary**: rappresenta il dizionario ricavato dal corpus processato. Per ciascun termine nel dizionario viene memorizzato anche il numero complessivo di documenti in cui compare il termine, e i puntatori a **Term Frequency data** e **Term Proximity data**, del termine.
- **Term Frequency data:** Per ciascun termine questa struttura contiene le informazioni riguardo ai documenti che contengono il termine, e la frequenza con cui compare nel documento.
- **Term Proximity data:** Per ciascun termine questa struttura permette di rappresentare le posizioni all'interno del documento in cui il termine compare.
- **Normalization factors:** Per ciascun campo in un documento vengono memorizzati dei fattori di normalizzazione.
- **Term Vectors**: Per ciascun documento e' possibilie memorizzare un vettore che contiene i termini contenuti nel documento con la relativa frequenza.

Struttura

- org.apache.lucene.document: contiene le classi per la rappresentazione del documento, attraverso i suoi campi.
- org.apache.lucene.analysis: contiene le classi per la gestione delle funzioni di pre-processing per l'indicizzazione del corpus.
- org.apache.lucene.index: contiene le classi per la rappresentazione dell'indice.
- org.apache.lucene.search: contiene le classi per la gestione delle funzioni che implementano differenti modelli di ricerca.
- org.apache.lucene.store: contiene le classi per la scrittura e lettura dell'indice su disco o altro.

analysis package

- Contiene le funzionalità corrispondenti al blocco *pre-processing* nello schema generale.
- Le principali funzionalità implementate sono:
  - Tokenizzazione.
  - Rimozione delle stop words
  - Rimozione dei caratteri accentati
  - Stemming (in differenti linguaggi)
    - Porter Stemmer (EN)
    - Snowball Stemmer (altri)

analysis package (2)

- Pregi:
  - Personalizzazione delle funzioni di tokenizzazione
  - Possibilità di caricare una lista di stopword personale
- Difetti:
  - Non offre nessuno strumento per l'individuazione di strutture complesse nella collezione. Ex. Un documento in una collezione può essere suddiviso in titolo, testo, autore.
  - Bisogna implementare di volta in volta il proprio sistema che genera la struttura del documento da indicizzare

search package

- La classe Query rappresenta l'astrazione delle funzionalità che permettono di esprimere un modello di IR. Le sue principali implementazioni in Lucene sono:
  - Term Query
  - Boolean Query
  - Wildcard Query
  - Phrase Query
  - Prefix Query
  - Fuzzy Query
  - Range Query
  - Span Query

Sintassi della query

- **Query Generica**
  - `pink panther`
- **Query a blocchi**
  - `"pink panther"`
- **Operatory booleani**
  - `"pink panther" AND return`
  - `"pink panther" +return`
- **Query per campi**
  - `title:"pink panther"`
- **Wildcard**
  - `pant?er`
  - `panther*`
- **Fuzzy**
  - `panther~`
  - `panther~0.8`
- **Pesature**
  - `pink panther^4`
- **Range**
  - `mod_date:[20070101 TO 20071001]`
- **Gruppi**
  - `title:(+return +"pink panther")`

Modello di ranking

- Lucene si basa sul TF * IDF all'interno di un Vector Space Model
- La funzione di similitudine tra un documento e la query è la seguente:

$$\text{sim}(q,d) = \text{coord}(q,d) \cdot \text{queryNorm}(q) \cdot \sum_{t \in q} \left(\text{tf}(t,q) \cdot \text{idf}(t)^2 \cdot \text{boost}(t) \cdot \text{norm}(t,d) \right)$$

Modello di ranking (2)

$$\mathsf{sim}(q,d) = \mathsf{coord}(q,d) \cdot \mathsf{queryNorm}(q) \cdot \sum_{t \in q} \left(\mathsf{tf}(t,d) \cdot \mathsf{idf}(t)^2 \cdot \mathsf{boost}(t) \cdot \mathsf{norm}(t,d) \right)$$

$\mathsf{coord}(q,d)$ Conta quanti termini di *q* compaiono nel documento *d*

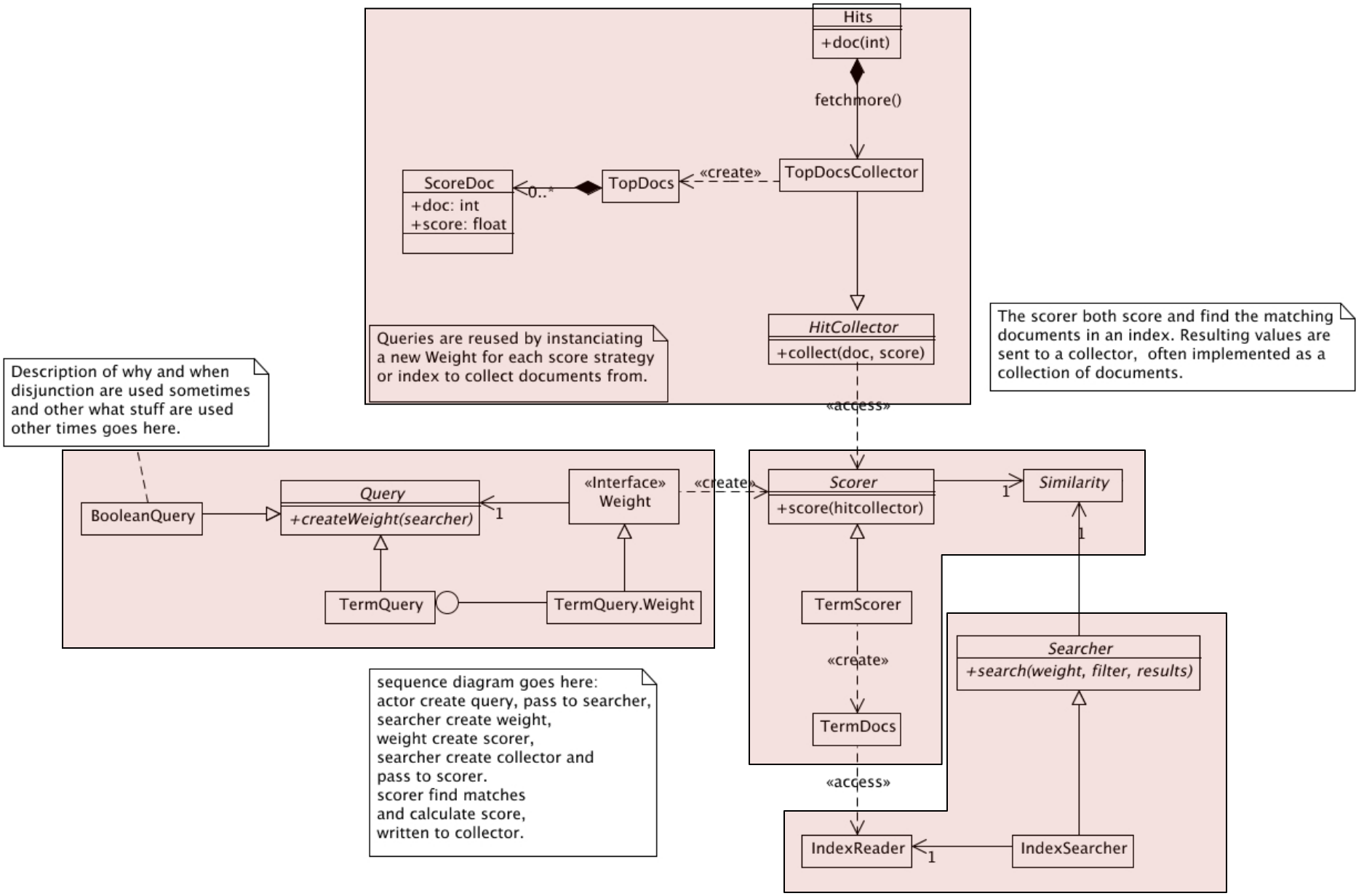
$\mathsf{boost}(t)$ Peso associato al termine o alla query

$\mathsf{tf}(t,q)$ $\sqrt{\#t_d}$

$\mathsf{idf}(t)$ $1 + \log\left(\frac{\#D}{\mathsf{docFreq}(t)}\right)$

$\mathsf{queryNorm}(q)$ $\frac{1}{\mathsf{boost}(q) \cdot \sum_{t \in q} (\mathsf{idf}(t) \cdot \mathsf{boost}(t))}$

$\mathsf{norm}(t,d)$ Combinazione del peso del *documento*, della lungezza e il peso del *campo*


Hits
+doc(int)
fetchmore()
ScoreDoc
+doc: int
+score: float
0..*
TopDocs
«create»
TopDocsCollector
HitCollector
+collect(doc, score)
Queries are reused by instanciating a new Weight for each score strategy or index to collect documents from.
The scorer both score and find the matching documents in an index. Resulting values are sent to a collector, often implemented as a collection of documents.
«access»
Description of why and when disjunction are used sometimes and other what stuff are used other times goes here.
BooleanQuery
Query
+createWeight(searcher)
1
«Interface»
Weight
«create»
Scorer
+score(hitcollector)
1
Similarity
1
TermQuery
TermQuery.Weight
TermScorer
«create»
TermDocs
Searcher
+search(weight, filter, results)
«access»
IndexReader
1
IndexSearcher
sequence diagram goes here:
actor create query, pass to searcher,
searcher create weight,
weight create scorer,
searcher create collector and
pass to scorer.
scorer find matches
and calculate score,
written to collector.

Schema procedurale per l'indicizzazione

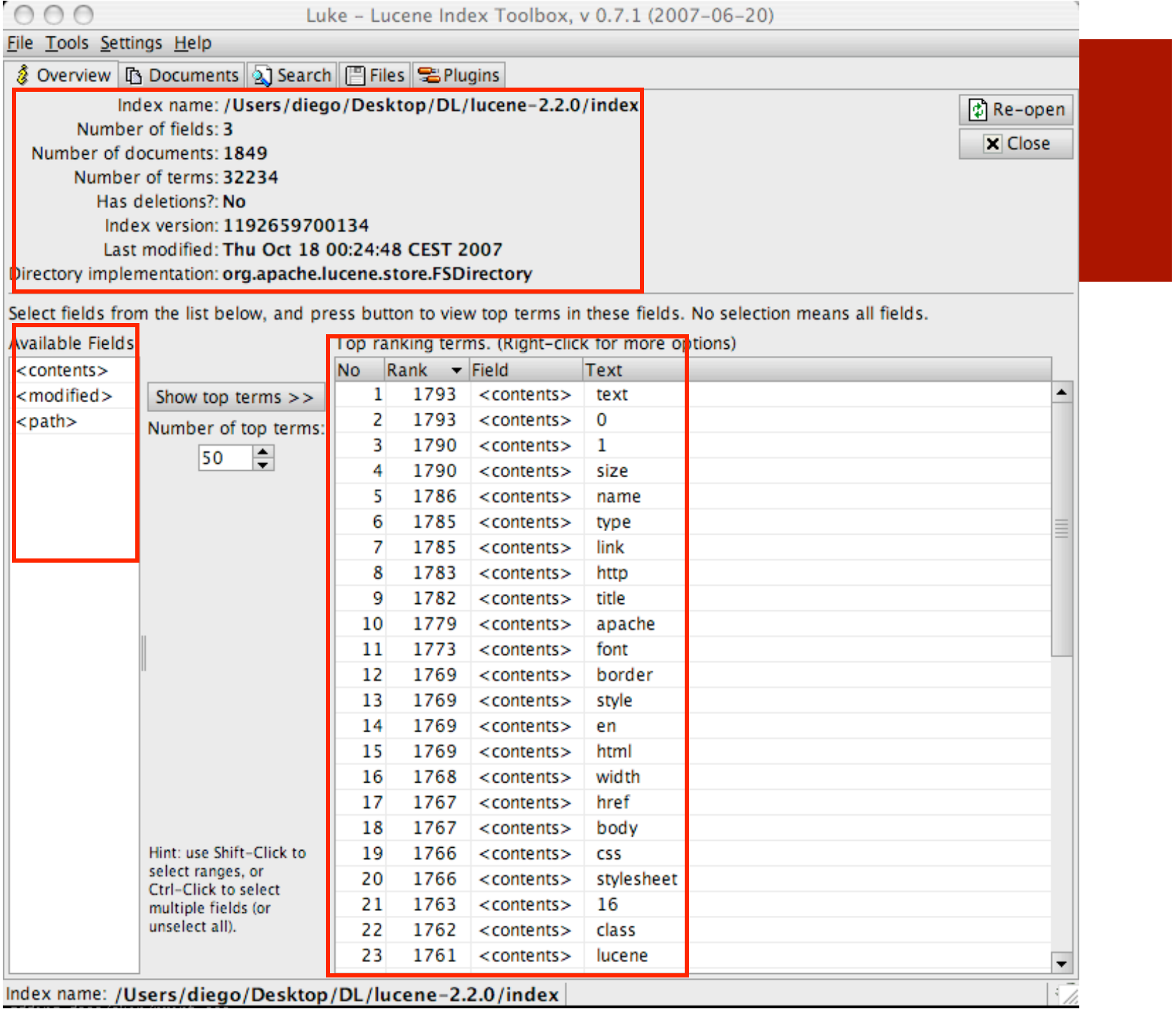
- Definire un modulo di pre-processamento attraverso la classe `Analyzer`;
- Per ciascun documento nella collezione: Creare un oggetto `Document` e aggiungere al documento i vari campi di tipo `Field`
- Creare un `IndexWriter` e aggiungere all'indice i vari campi tramite il metodo `addDocument()`;

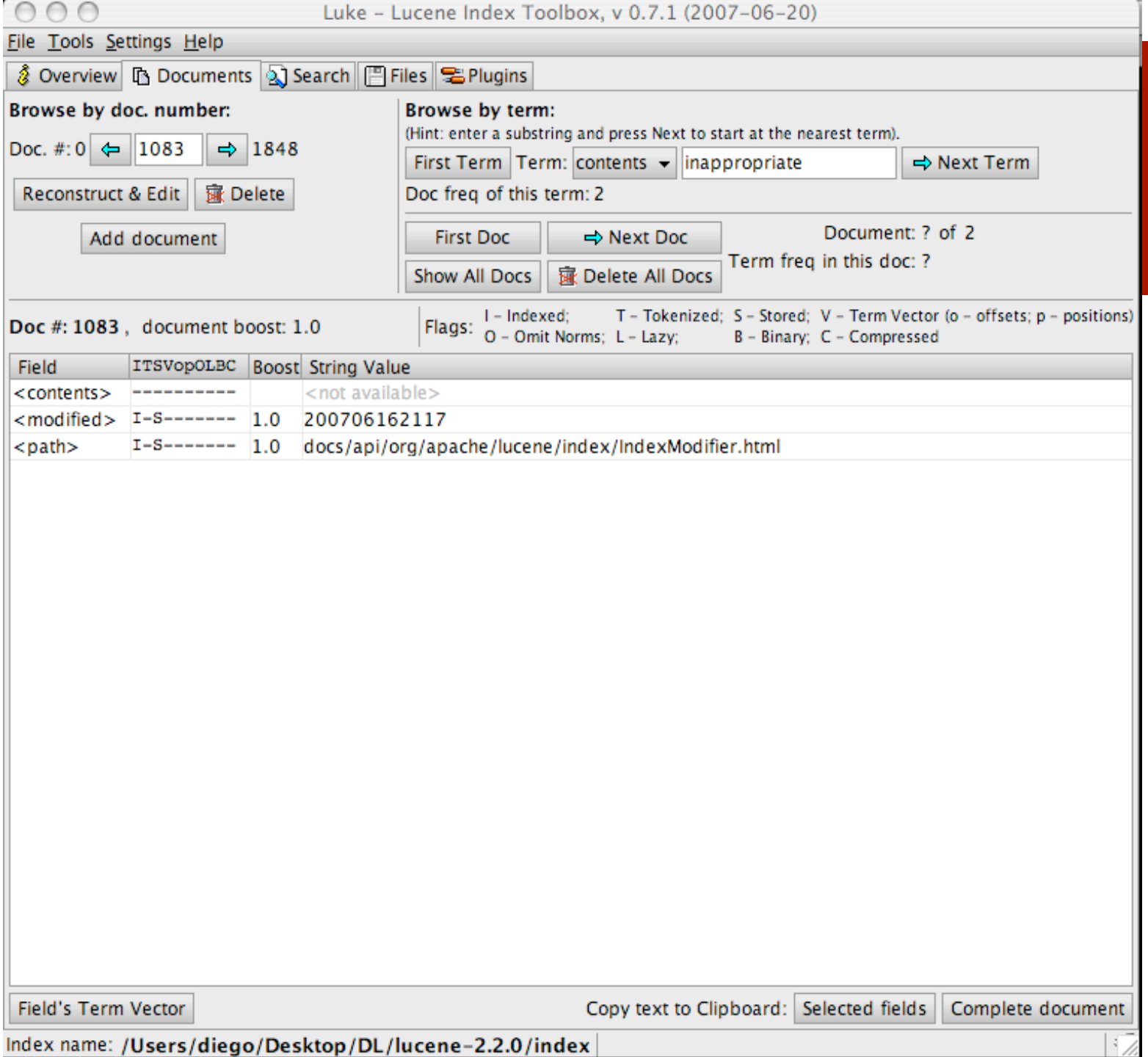
Schema procedurale per la ricerca

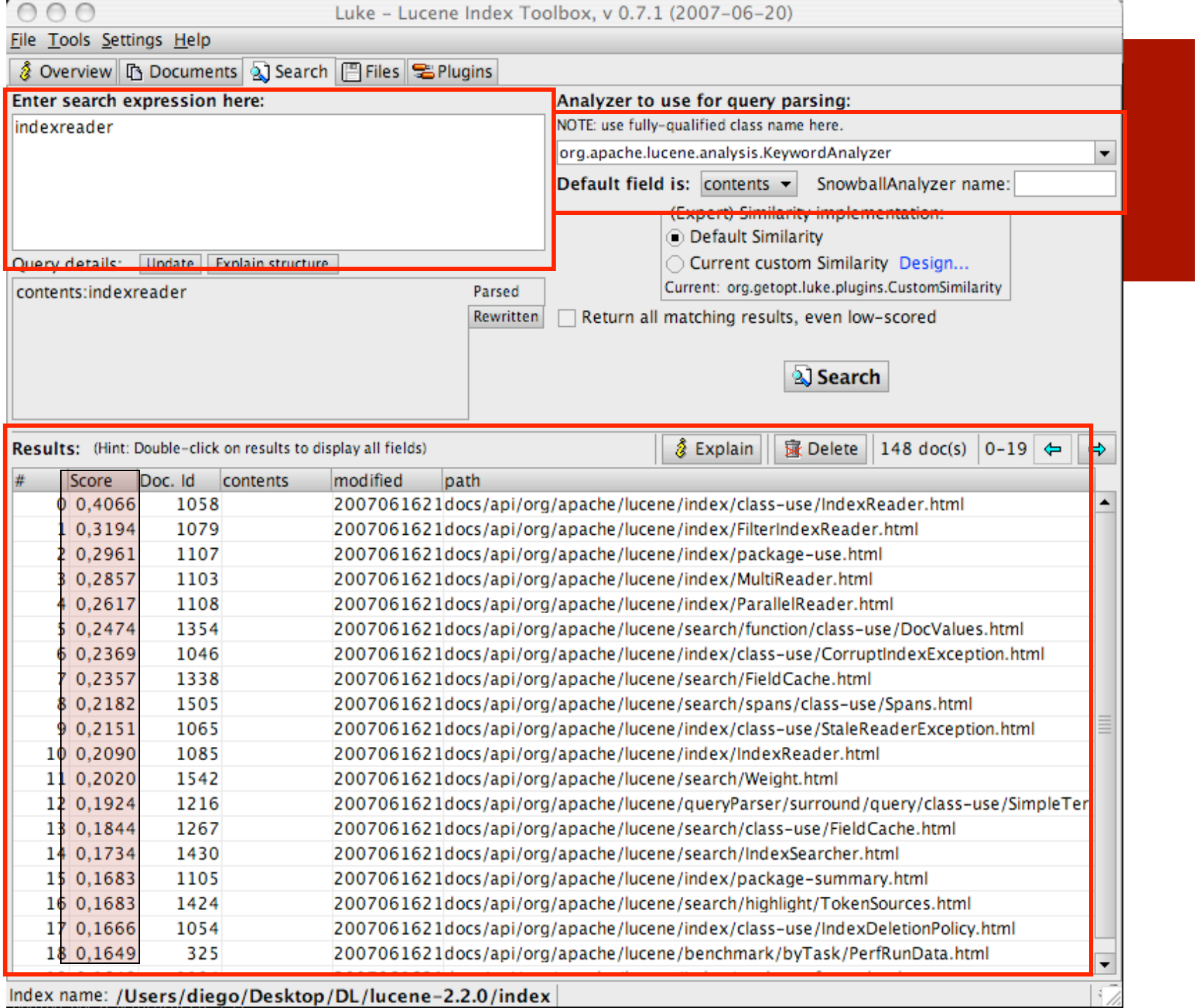
- Definire lo stesso modulo di pre-processamento usato durante l'indicizzazione attraverso la classe `Analyzer`;
- Aprire l'indice dei documenti attraverso la classe `IndexSearcher`
- Definire il parser della query (`QueryParser`)
- Definire il modello di `Query` che si vuole adottare
- Ricercare i documenti secondo il modello scelto attraverso la classe `IndexSearcher`.

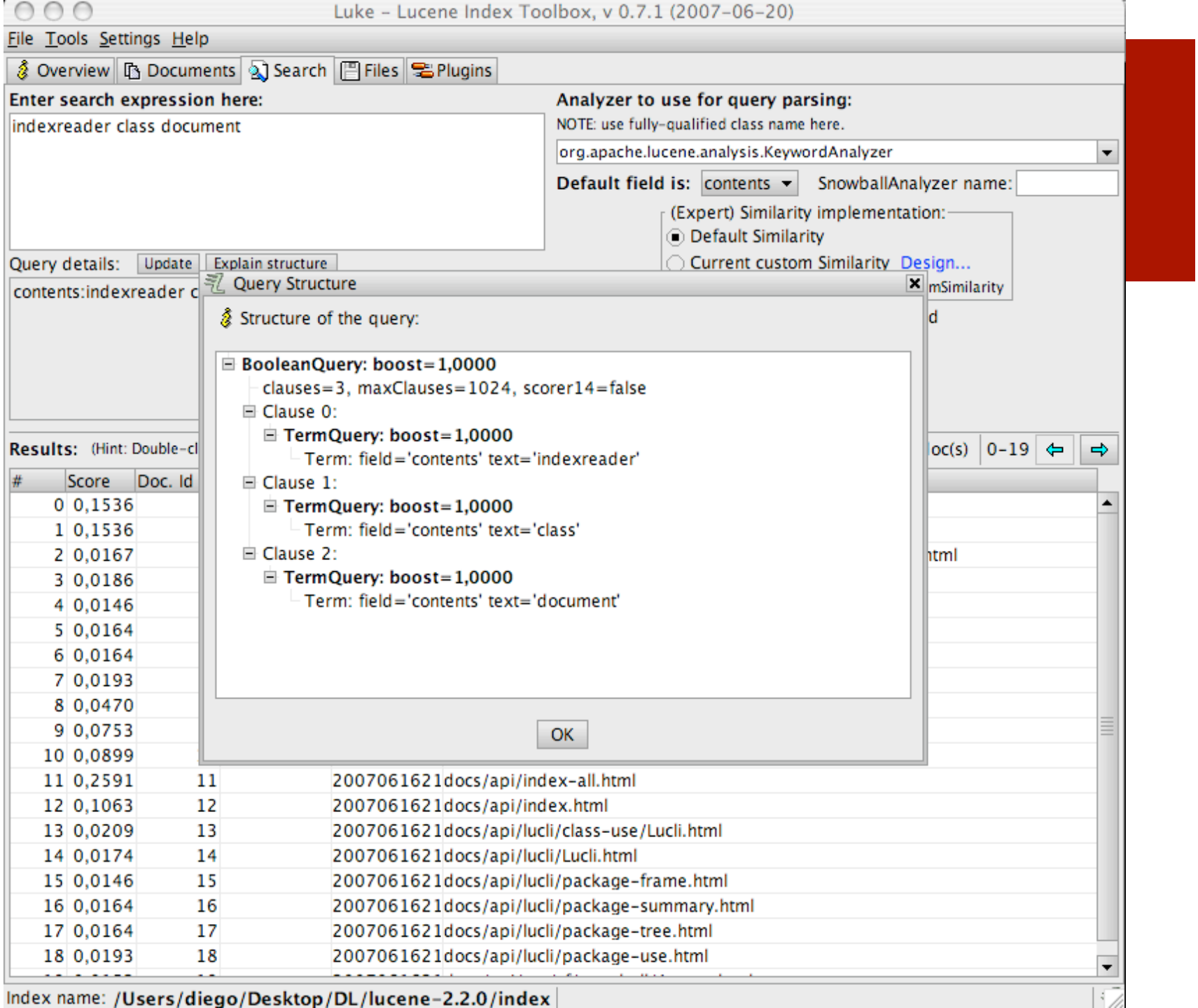
Interfaccia grafica: Luke

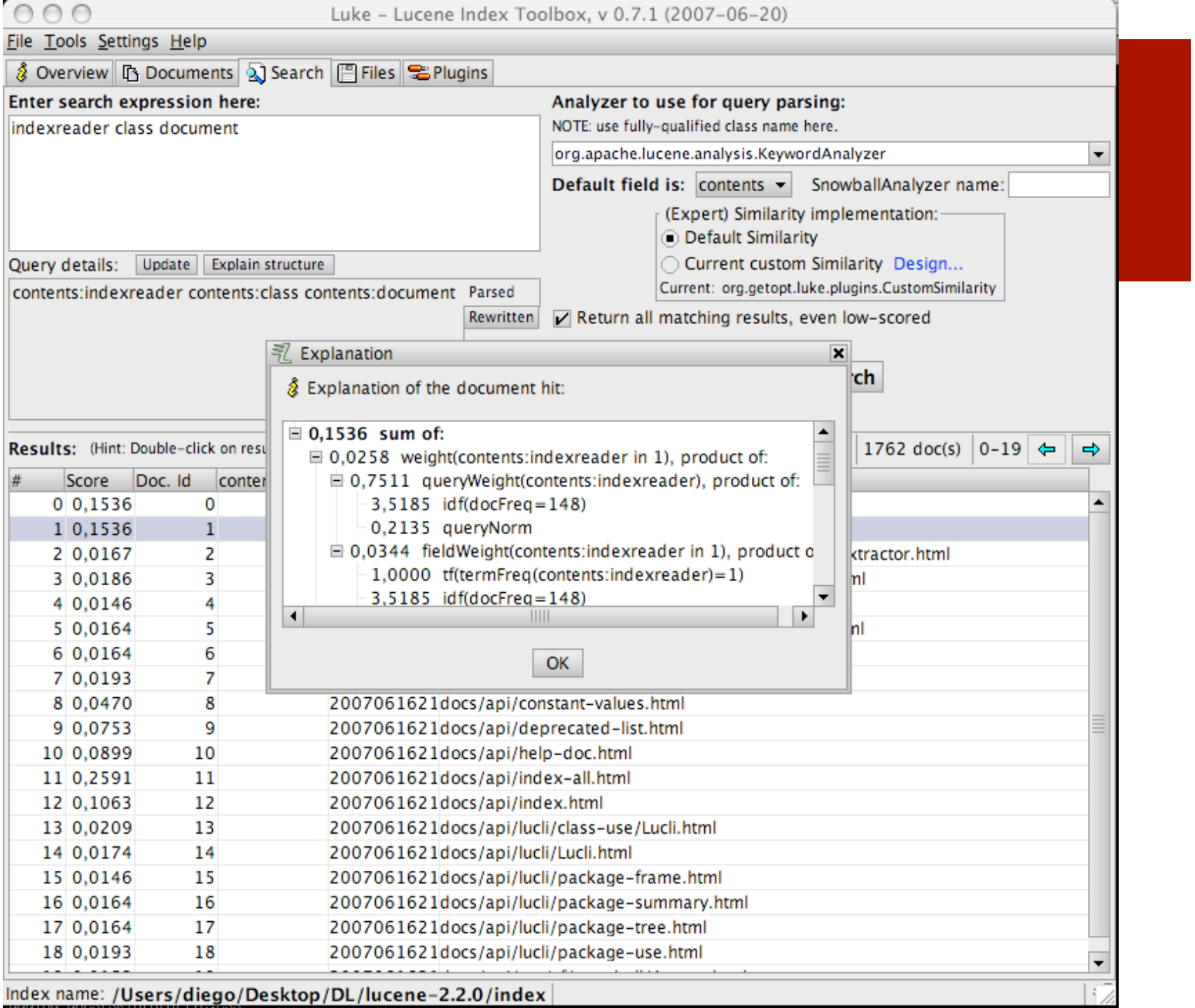
- **Luke** è un interfaccia grafica che permette di accedere agli indici creati da lucene ed eseguire una serie di operazioni:
  - Vedere i documenti indicizzati con i rispettivi campi
  - Eseguire e testare delle query complesse
  - Modificare l'indice cancellando documenti o modificando i dati già indicizzati.
  - E molto altro …
- La potete scaricare da: http://www.getopt.org/luke/

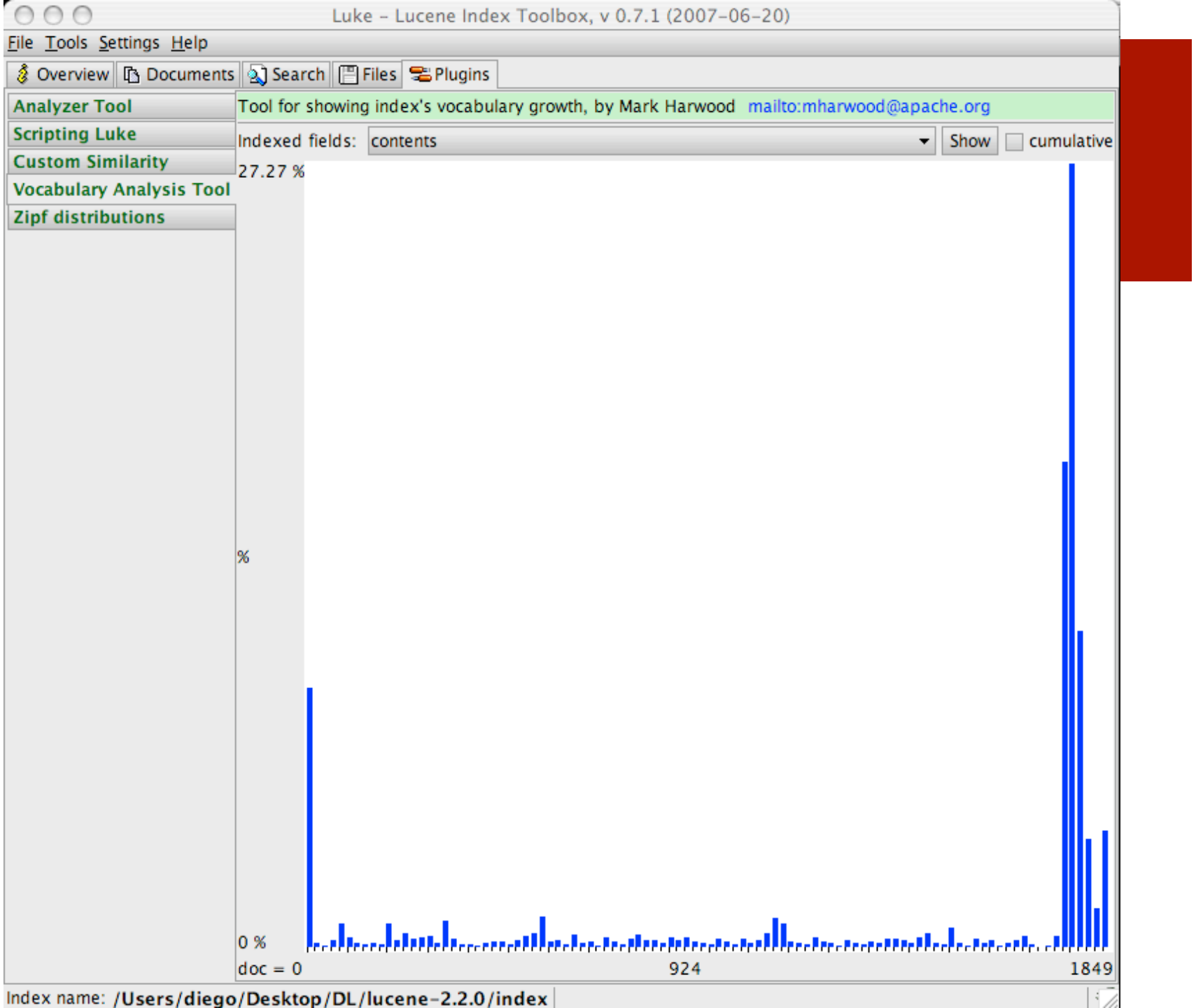

Luke - Lucene Index Toolbox, v 0.7.1 (2007-06-20)
File Tools Settings Help
Overview Documents Search Files Plugins
Index name: /Users/diego/Desktop/DL/lucene-2.2.0/index
Number of fields: 3
Number of documents: 1849
Number of terms: 32234
Has deletions?: No
Index version: 1192659700134
Last modified: Thu Oct 18 00:24:48 CEST 2007
Directory implementation: org.apache.lucene.store.FSDirectory
Re-open
Close
Select fields from the list below, and press button to view top terms in these fields. No selection means all fields.
Available Fields
<contents>
<modified>
<path>
Show top terms >>
Number of top terms:
50
Hint: use Shift-Click to select ranges, or Ctrl-Click to select multiple fields (or unselect all).
Top ranking terms. (Right-click for more options)
No Rank Field Text
1 1793 <contents> text
2 1793 <contents> 0
3 1790 <contents> 1
4 1790 <contents> size
5 1786 <contents> name
6 1785 <contents> type
7 1785 <contents> link
8 1783 <contents> http
9 1782 <contents> title
10 1779 <contents> apache
11 1773 <contents> font
12 1769 <contents> border
13 1769 <contents> style
14 1769 <contents> en
15 1769 <contents> html
16 1768 <contents> width
17 1767 <contents> href
18 1767 <contents> body
19 1766 <contents> css
20 1766 <contents> stylesheet
21 1763 <contents> 16
22 1762 <contents> class
23 1761 <contents> lucene
Index name: /Users/diego/Desktop/DL/lucene-2.2.0/index


Luke – Lucene Index Toolbox, v 0.7.1 (2007-06-20)
File Tools Settings Help
Overview Documents Search Files Plugins
Browse by doc. number:
Doc. #: 0 1083 1848
Reconstruct & Edit Delete
Add document
Browse by term:
(Hint: enter a substring and press Next to start at the nearest term).
First Term Term: contents inappropriate Next Term
Doc freq of this term: 2
First Doc Next Doc
Show All Docs Delete All Docs
Document: ? of 2
Term freq in this doc: ?
Doc #: 1083 , document boost: 1.0
Flags: I – Indexed; T – Tokenized; S – Stored; V – Term Vector (o – offsets; p – positions)
O – Omit Norms; L – Lazy; B – Binary; C – Compressed
Field ITSVopOLBC Boost String Value
<contents> ---------- <not available>
<modified> I-S------- 1.0 200706162117
<path> I-S------- 1.0 docs/api/org/apache/lucene/index/IndexModifier.html
Field's Term Vector
Copy text to Clipboard: Selected fields Complete document
Index name: /Users/diego/Desktop/DL/lucene-2.2.0/index


Luke - Lucene Index Toolbox, v 0.7.1 (2007-06-20)
File Tools Settings Help
Overview Documents Search Files Plugins
Enter search expression here:
indexreader
Query details: Update Explain structure
contents:indexreader
Parsed
Rewritten
Analyzer to use for query parsing:
NOTE: use fully-qualified class name here.
org.apache.lucene.analysis.KeywordAnalyzer
Default field is: contents
SnowballAnalyzer name:
(Expert) Similarity implementation:
Default Similarity
Current custom Similarity Design...
Current: org.getopt.luke.plugins.CustomSimilarity
Return all matching results, even low-scored
Search
Results: (Hint: Double-click on results to display all fields)
Explain
Delete
148 doc(s)
0-19
Score Doc. Id contents modified path
0 0,4066 1058 20070616 21 docs/api/org/apache/lucene/index/class-use/IndexReader.html
1 0,3194 1079 20070616 21 docs/api/org/apache/lucene/index/FilterIndexReader.html
2 0,2961 1107 20070616 21 docs/api/org/apache/lucene/index/package-use.html
3 0,2857 1103 20070616 21 docs/api/org/apache/lucene/index/MultiReader.html
4 0,2617 1108 20070616 21 docs/api/org/apache/lucene/index/ParallelReader.html
5 0,2474 1354 20070616 21 docs/api/org/apache/lucene/search/function/class-use/DocValues.html
6 0,2369 1046 20070616 21 docs/api/org/apache/lucene/index/class-use/CorruptIndexException.html
7 0,2357 1338 20070616 21 docs/api/org/apache/lucene/search/FieldCache.html
8 0,2182 1505 20070616 21 docs/api/org/apache/lucene/search/spans/class-use/Spans.html
9 0,2151 1065 20070616 21 docs/api/org/apache/lucene/index/class-use/StaleReaderException.html
10 0,2090 1085 20070616 21 docs/api/org/apache/lucene/index/IndexReader.html
11 0,2020 1542 20070616 21 docs/api/org/apache/lucene/search/Weight.html
12 0,1924 1216 20070616 21 docs/api/org/apache/lucene/queryParser/surround/query/class-use/SimpleTer
13 0,1844 1267 20070616 21 docs/api/org/apache/lucene/search/class-use/FieldCache.html
14 0,1734 1430 20070616 21 docs/api/org/apache/lucene/search/IndexSearcher.html
15 0,1683 1105 20070616 21 docs/api/org/apache/lucene/index/package-summary.html
16 0,1683 1424 20070616 21 docs/api/org/apache/lucene/search/highlight/TokenSources.html
17 0,1666 1054 20070616 21 docs/api/org/apache/lucene/index/class-use/IndexDeletionPolicy.html
18 0,1649 325 20070616 21 docs/api/org/apache/lucene/benchmark/byTask/PerfRunData.html
Index name: /Users/diego/Desktop/DL/lucene-2.2.0/index


Luke - Lucene Index Toolbox, v 0.7.1 (2007-06-20)
File Tools Settings Help
Overview Documents Search Files Plugins
Enter search expression here:
indexreader class document
Analyzer to use for query parsing:
NOTE: use fully-qualified class name here.
org.apache.lucene.analysis.KeywordAnalyzer
Default field is: contents
SnowballAnalyzer name:
(Expert) Similarity implementation:
Default Similarity
Current custom Similarity
Design...
Query details:
Update
Explain structure
contents:indexreader c
Query Structure
Structure of the query:
BooleanQuery: boost=1,0000
clauses=3, maxClauses=1024, scorer14=false
Clause 0:
TermQuery: boost=1,0000
Term: field='contents' text='indexreader'
Clause 1:
TermQuery: boost=1,0000
Term: field='contents' text='class'
Clause 2:
TermQuery: boost=1,0000
Term: field='contents' text='document'
OK
Results:
0-19
Score Doc. Id
0 0,1536
1 0,1536
2 0,0167
3 0,0186
4 0,0146
5 0,0164
6 0,0164
7 0,0193
8 0,0470
9 0,0753
10 0,0899
11 0,2591 11 2007061621docs/api/index-all.html
12 0,1063 12 2007061621docs/api/index.html
13 0,0209 13 2007061621docs/api/lucli/class-use/Lucli.html
14 0,0174 14 2007061621docs/api/lucli/Lucli.html
15 0,0146 15 2007061621docs/api/lucli/package-frame.html
16 0,0164 16 2007061621docs/api/lucli/package-summary.html
17 0,0164 17 2007061621docs/api/lucli/package-tree.html
18 0,0193 18 2007061621docs/api/lucli/package-use.html
Index name: /Users/diego/Desktop/DL/lucene-2.2.0/index


Luke - Lucene Index Toolbox, v 0.7.1 (2007-06-20)
File Tools Settings Help
Overview Documents Search Files Plugins
Enter search expression here:
indexreader class document
Analyzer to use for query parsing:
NOTE: use fully-qualified class name here.
org.apache.lucene.analysis.KeywordAnalyzer
Default field is: contents
SnowballAnalyzer name:
(Expert) Similarity implementation:
Default Similarity
Current custom Similarity Design...
Current: org.getopt.luke.plugins.CustomSimilarity
Query details: Update Explain structure
contents:indexreader contents:class contents:document Parsed
Rewritten
Return all matching results, even low-scored
Explanation
Explanation of the document hit:
0,1536 sum of:
0,0258 weight(contents:indexreader in 1), product of:
0,7511 queryWeight(contents:indexreader), product of:
3,5185 idf(docFreq=148)
0,2135 queryNorm
0,0344 fieldWeight(contents:indexreader in 1), product o
1,0000 tf(termFreq(contents:indexreader)=1)
3,5185 idf(docFreq=148)
OK
Results: (Hint: Double-click on resu
1762 doc(s) 0-19
Score Doc. Id conter
0 0,1536 0
1 0,1536 1
2 0,0167 2 xtractor.html
3 0,0186 3 ml
4 0,0146 4
5 0,0164 5 ml
6 0,0164 6
7 0,0193 7
8 0,0470 8 2007061621docs/api/constant-values.html
9 0,0753 9 2007061621docs/api/deprecated-list.html
10 0,0899 10 2007061621docs/api/help-doc.html
11 0,2591 11 2007061621docs/api/index-all.html
12 0,1063 12 2007061621docs/api/index.html
13 0,0209 13 2007061621docs/api/lucli/class-use/Lucli.html
14 0,0174 14 2007061621docs/api/lucli/Lucli.html
15 0,0146 15 2007061621docs/api/lucli/package-frame.html
16 0,0164 16 2007061621docs/api/lucli/package-summary.html
17 0,0164 17 2007061621docs/api/lucli/package-tree.html
18 0,0193 18 2007061621docs/api/lucli/package-use.html
Index name: /Users/diego/Desktop/DL/lucene-2.2.0/index


Luke - Lucene Index Toolbox, v 0.7.1 (2007-06-20)
File Tools Settings Help
Overview Documents Search Files Plugins
Analyzer Tool
Scripting Luke
Custom Similarity
Vocabulary Analysis Tool
Zipf distributions
Tool for showing index's vocabulary growth, by Mark Harwood mailto:mharwood@apache.org
Indexed fields: contents
Show
cumulative
27.27 %
%
0 %
doc = 0
924
1849
Index name: /Users/diego/Desktop/DL/lucene-2.2.0/index

Lucene - qualche numero

- Occupazione del disco minima: circa il 20% - 30% della dimensione del corpus originale
- Il numero massimo di documenti / termini indicizzabili sono pari al massimo numero di intero rappresentabile con 32bit
- 20Mb full text vengono indicizzati in circa un minuto su un Pentium M 1.5GHz
- Minimo uso della ram ~1MB heap

Solr

- Implementazione da parte di Apache di un enterprise search server standalone sopra le API di Lucene
- Fornisce API per l'accesso REST-like
- Supporta differenti interfacce di comunicazione (JSON, XML)
- Ha un interfaccia di amministrazione web
- Supporta la distribuzione e la replica degli indici
- http://lucene.apache.org/solr/

Lemur

- Lemur è un altro toolkit open-source
  - permette di avere velocemente un sistema di IR.
- E' realizzato attraverso una collaborazione tra il ***Computer Science Department at the University of Massachusetts*** e la ***School of Computer Science at Carnegie Mellon University***.
- http://www.lemurproject.org/
- http://sourceforge.net/p/lemur/wiki/Home/

Lemur - caratteristiche principali

- Lemur integra diverse funzionalità per indicizzare una grande varietà di formati:
  - TREC text/web format
  - XML/HTML documents
  - Mbox (unix mailbox file)
  - MS Office documents (doc, ppt)
  - Pdf documents
- Il core e' scritto in C/C++, l'interfaccia grafica in Java.

Lemur

- Pro:
  - Possibilità di indicizzare metadati
    - EX: Nel <year> 2009 </year> si tiene il corso di <course> Web Mining and Rerieval </course>
  - Possibilità di eseguire query complesse sui metadati
    - #between(year 2008 2010)
  - Diversi modelli di pesatura già implementati
    - tf.idf, Okapi and InQuery
- Contro:
  - Scarsa possibilità di personalizzare campi e/o documenti strutturati.

Lemur - Tutorial

- **Lemur Toolkit Tutorial**
  - http://www.lemurproject.org/tutorials/lemur_sigir_2006.ppt
- An Overview of the **Indri Search Engine**
  - http://ciir.cs.umass.edu/~metzler/presentations/uiuc-indri.ppt

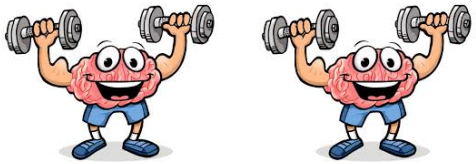
Collezione di test

- Cranfield collection 1398 abstracts (Aereodinamica) scaricabile dal sito del corso

```
.I 1
.T
titolo
.A
autori
.B
riferimenti bibliografici
.W
testo
```

Esempio

- Implementare un metodo di indicizzazione che sfrutti le strutture del testo originale (titolo, testo, etc.)
- Implementare un metodo di ricerca che sfrutti la struttura del corpus
  - Per ricercare un insieme di termini sia nel titolo che nel corpo del testo dando maggiore importanza ai termini presenti nel titolo



Esercizio

- Implementare un valutatore di prestazioni rispetto alla Cranfield Collection
  - Dato il file delle query (cran_query.text)
  - E il file dei risultati rilevanti (qrels.txt)
- Calcolare
  - Mean Average Precision del sistema rispetto alle query fornite

$$MAP(Q) = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{|R_q|} \sum_{d \in R_q} P@k_{q,d}$$

Q = set of queries

R_q=set of relevant documents for the query q

$K_{q,d}$=ranking of the document d retrieved throught the query q

Mean Average Precision

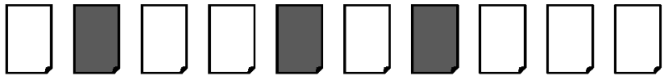

= relevant documents for query 1

Ranking #1

Recall	0.2	0.2	0.4	0.4	0.4	0.6	0.6	0.6	0.8	1.0
Precision	1.0	0.5	0.67	0.5	0.4	0.5	0.43	0.38	0.44	0.5


= relevant documents for query 2

Ranking #2



Recall	0.0	0.33	0.33	0.33	0.67	0.67	1.0	1.0	1.0	1.0
Precision	0.0	0.5	0.33	0.25	0.4	0.33	0.43	0.38	0.33	0.3

$$average\ precision\ query\ 1 = (1.0 + 0.67 + 0.5 + 0.44 + 0.5)/5 = 0.62$$
$$average\ precision\ query\ 2 = (0.5 + 0.4 + 0.43)/3 = 0.44$$

$$mean\ average\ precision = (0.62 + 0.44)/2 = 0.53$$

Esercizio

- Implementare un metodo di pseudo relevance feedback aggiungendo i termini del titolo dei primi *m* documenti alla query con un peso pari alla metà dei termini della query originale
- Verificare l'impatto sulla *Mean Average Precision* di questa tecnica