Tracce dei progetti

Progetto 1: Indicizzazione ed interrogazione con Lucene

1) Formare gruppi di 3 studenti
2) Selezionare 10 documenti su un argomento a piacere
3) Indicizzare i documenti con Lucene
4) Definire 4 task di retrieval (individuando alcuni specifici bisogni informativi)
5) Per ogni task *T*, valutare la rilevanza di ogni documento rispetto a *T*:
   - identificare i documenti rilevanti per *T* ed i documenti non rilevanti per *T*
   - ogni membro del gruppo esprime il suo giudizio sulla rilevanza dei documenti
6) Per ogni task di retrieval, formulare 2 query che permettano di risolverlo:
   - un'espressione booleana
   - un insieme di termini (eventualmente pesato)
7) Eseguire le query con Lucene
8) Calcolare le misure di *recall* e *precision* per tutte le query eseguite
   - Nel caso in cui il risultato sia una lista ordinata (query, disegnare le curve recall-precision.
9) Scrivere una relazione sul lavoro svolto

Progetto 1: come costruire le curve recall-precision

Risultato della query = lista ordinata

abcdef	NO
xvxcds	**OK**
Adihui	NO
Likvh	NO
Jff	**OK**
Fgv	NO
Fgrer	NO
Gdgv	NO
Dgh	NO
Ghkl:i	**OK**
Loikih	NO
Jlki	NO
Sasd	NO
Dff	**OK**
V4vcv	NO
Sfsfgd33	NO
Lki.h	NO
Lkp:i	NO
.liiy	**OK**

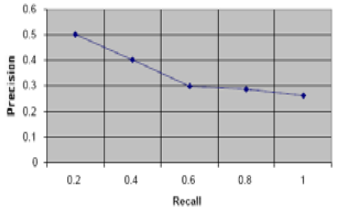
Supponiamo che i (5) documenti rilevanti per la query appaiono nelle posizioni 2,5,10,14 e 19

Fissando i livelli di recall a

1/5, 2/5, 3/5, 4/5, 5/5

i relativi valori di precision sono

1/2, 2/5, 3/10, 4/14, 5/19



Progetto 2: Costruzione di un DB per simulare il calcolo di query nel modello vettoriale

INPUT: tre file di testo (formato cvs: valori separati da ";")

"docs.csv"

doc_id;term_id;tf;stemmed_word
1;297;1;span
1;1054;2;part
1;1373;1;bas
…

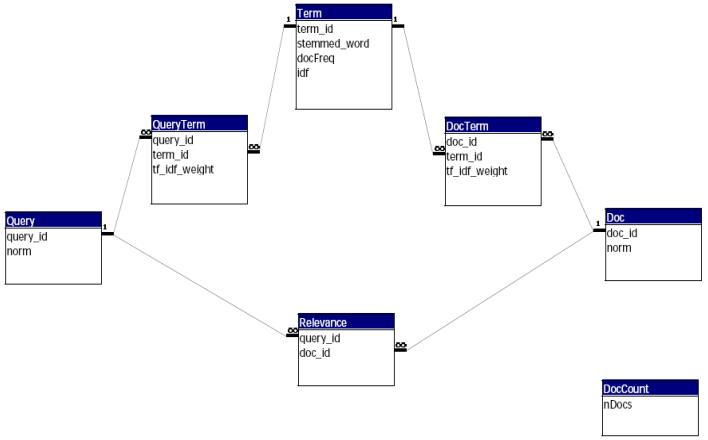
"queries.csv"

query_id;term_id;tf;stemmed_word
1;356;1;spee
1;2320;1;aeroelast
1;2726;1;obey
…

"relevance.csv"

query_id;doc_id;
1;184;
1;29;
1;31;

Lo schema del DB da costruire



Passi per costruire ed usare il DB

1. Importazione dei tre file in tre tabelle distinte
   - docs.csv → *xDocTerm* (**temporanea**)
   - queries.csv → *xQueryTerm* (**temporanea**)
   - relevance.csv → *Relevance*
2. Popolamento delle varie tabelle, mediante query SQL
   1. *DocCount*, da *xDocTerm* (… conto i documenti)
   2. *Term*, da *DocCount* e *xDocTerm* (… calcolo idf di ogni termine)
   3. DocTerm, da *Term* e *xDocTerm*) (… calcolo i pesi tf-idf)
   4. *QueryTerm,* da *Term* e *xQueryTerm* (… calcolo i pesi tf-idf)
   5. *Doc*, da *DocTerm* (… calcolo la norma di ogni documento)
   6. *Query*, da *QueryTerm* (… calcolo la norma di ogni query)
3. Scrittura di un'istruzione SQL per il ranking dei documenti rispetto ad una data query (… similarità del coseno fra ogni documento e la query)

Popolamento tabella *DocCount*

```
SELECT COUNT(*) AS nDocs
INTO DocCount
FROM
( SELECT DISTINCT doc_id FROM xDocTerm );
```

Popolamento della tabella *Term*

```
SELECT xDocTerm.term_id, xDocTerm.stemmed_word,
        COUNT(*) AS docFreq,
        LOG( MAX(DocCount.nDocs)/COUNT(*) ) / LOG(2) + 1 AS idf
INTO Term
FROM xDocTerm, DocCount
GROUP BY xDocTerm.term_id, xDocTerm.stemmed_word;
```

$$\text{Log}_2 (N/n_i) = \text{Log} (N/n_i) / \text{Log}(2)$$

Popolamento della tabella *DocTerm*

```
SELECT DISTINCT xDocTerm.doc_id, xDocTerm.term_id,
        (xDocTerm.tf_weight * Term.idf) AS tf_idf_weight
INTO DocTerm
FROM xDocTerm, Term
WHERE Term.term_id = xDocTerm.term_id
ORDER BY xDocTerm.doc_id, xDocTerm.term_id;
```

Popolamento della tabella *QueryTerm*

```
SELECT DISTINCT xQueryTerm.query_id,
        xQueryTerm.term_id,
        (xQueryTerm.tf_weight * Term.idf ) AS tf_idf_weight
INTO QueryTerm
FROM xQueryTerm, Term
WHERE Term.term_id= xQueryTerm.term_id
ORDER BY xQueryTerm.query_id, xQueryTerm.term_id;
```

Popolamento della tabella *Doc*

```
SELECT DISTINCT DocTerm.doc_id,
        SQR(
            SUM( DocTerm.tf_idf_weight * DocTerm.tf_idf_weight )
        )
        AS norm
INTO Doc
FROM DocTerm
GROUP BY DocTerm.doc_id;
```

Popolamento della tabella *Query*

```
SELECT DISTINCT query_id,
        SQR( SUM( tf_idf_weight * tf_idf_weight ) ) AS norm
INTO Query
FROM QueryTerm
GROUP BY query_id;
```

Query per il ranking dei documenti rispetto ad una data query

```
SELECT Doc.doc_id, Doc.norm AS norm,
        SQR(
           SUM( DocTerm.tf_idf_weight * QueryTerm.tf_idf_weight )
        )
        / Doc.norm
        AS      Cos_Sim
FROM DocTerm, Doc, QueryTerm
WHERE DocTerm.term_id=QueryTerm.term_id
        AND Doc.doc_id=DocTerm.doc_id
        AND QueryTerm.query_id=[idQuery]
GROUP BY Doc.doc_id, Doc.norm;
```