Outils pour la RI

EARIA 2014

16 octobre 2014

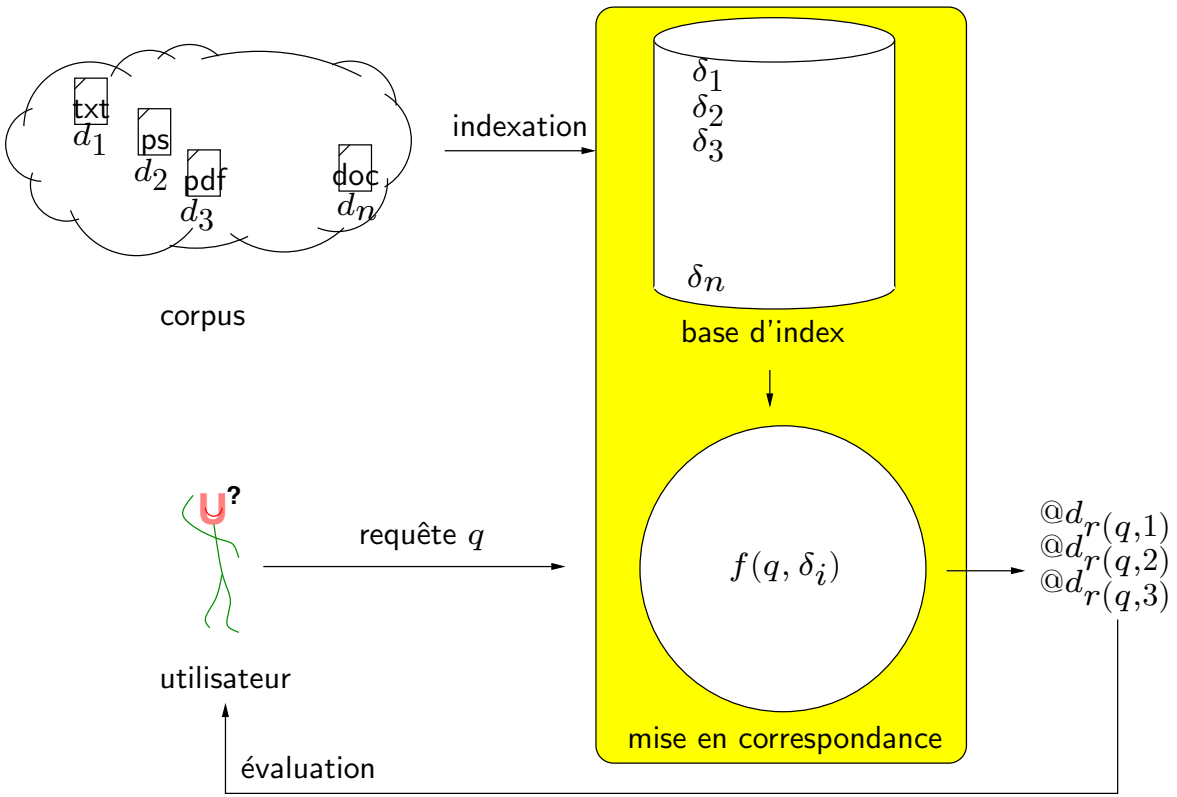
Michel Beigbeder

École Nationale Supérieure des Mines de Saint-Étienne
mbeig@emse.fr

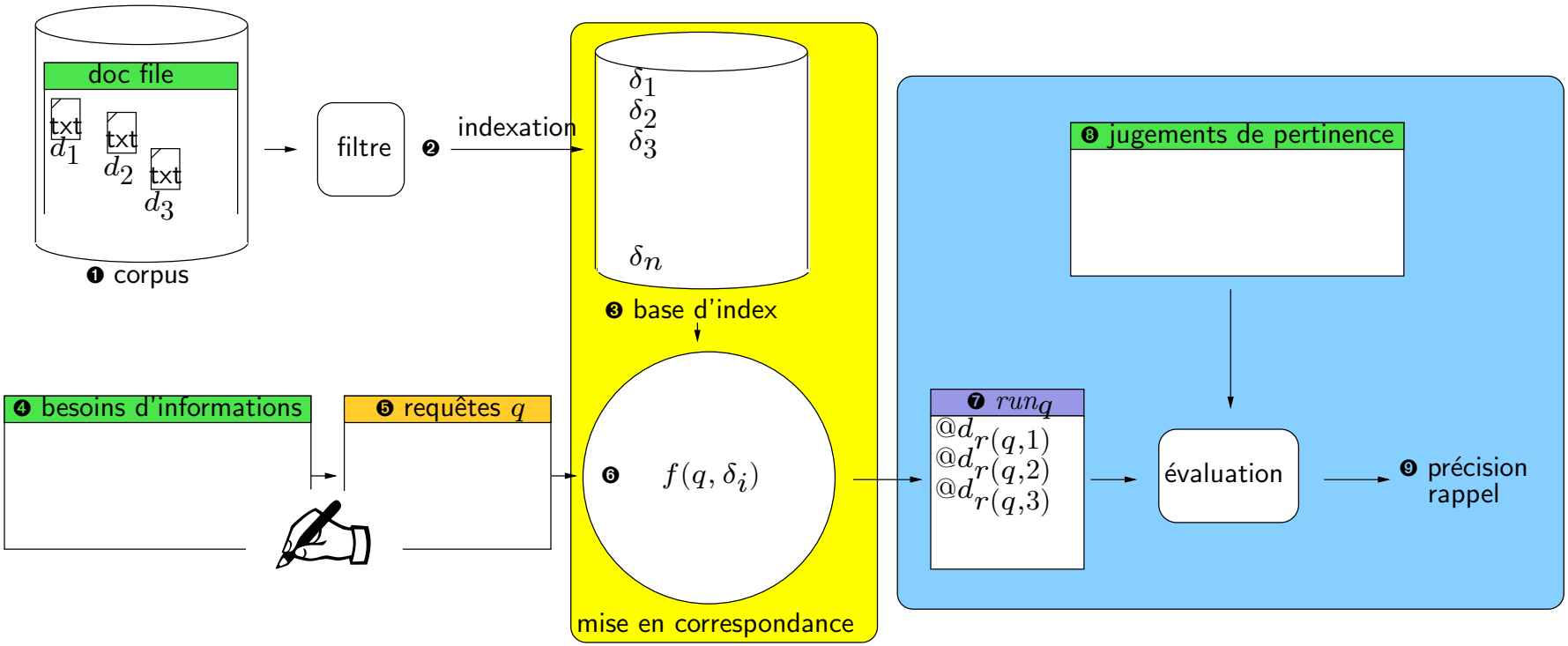
Recherche d'information : les tâches

- Recherche documentaire ($ad'hoc$)
- Routage et filtrage
- Extraction d'information
- Classification automatique (supervisée ou non)
- Résumé
- Recommandation
- Systèmes question-réponse

Modèle de RI


txt
d_1
ps
d_2
pdf
d_3
doc
d_n
corpus
indexation
δ_1
δ_2
δ_3
δ_n
base d'index
$f(q, \delta_i)$
mise en correspondance
requête q
utilisateur
évaluation
$@d_r(q,1)$
$@d_r(q,2)$
$@d_r(q,3)$

Modèle TREC de RI (1/2)



Modèle TREC de RI (2/2)

1. documents originaux
2. fichier(s) indexable(s)
3. index des documents
4. les besoins d'informations
5. les requêtes
6. les index des requêtes
7. les listes de réponses retournées par le moteur
8. les ensembles de documents jugés pertinents
9. l'évaluation précision-rappel

❶L'ensemble de documents

- Collecte
  - Un fichier pour plein de documents (formats SMART, TREC)
    Collections de tests
  - Une source de données et des critères (par exemple un répertoire et une liste de suffixes)
    Recherche sur un poste de travail (*Desktop search*)
  - Parcours par URL (fichiers HTML et pdf)
    Recherche dans un site ou sur le Web (*Site search, Web search*)
  - Par dépôt des documents

Extraits de collections

adi.all	WT10G	INEX Wikipedia
.I 1 .T the ibm dsd technical information cen combining traditional library feature and mechanized computer processing .A H. S. WHITE .W the ibm data systems division technic information center (tic) provides an system for integrated and compatible processing of technical information the system offers several advantage 1 . it is a sophisticated mechan and retrieval; 2 . it is compatible with all li records produced under a standard p within ibm libraries, providing suc as 3 x 5 catalog cards, circulation notices; 3 . it is reversible, so that di processing would not cause gaps in th manual records;	<DOC> <DOCNO>WTX001-B19-1</DOCNO> <DOCOLDNO>IA001-000003-B014-86</DOCOL <DOCHDR> http://seamonkey.ed.asu.edu:80/oz/ 12 2963 HTTP/1.0 200 OK Server: Netscape-Communications/1.1 Date: Wednesday, 01-Jan-97 03:36:14 G Last-modified: Thursday, 04-Jul-96 06 Content-length: 2770 Content-type: text/html </DOCHDR> <html <head> <title> An Oz Home Page</title></head <BODY BGCOLOR="#abcdef" TEXT="#000000 ALINK="#FDF5E6"> <h1 align=center>The Seamonkey OZ Hom <IMG SRC="lin_rain.gif" width="100%"> <h2 align=center>The OZ Home Page is Page.	<?xml version="1.0" encoding="UTF-8"? <!-- generated by CLiX/Wiki2XML [MPI- <!DOCTYPE article SYSTEM "../article. <article xmlns:xlink="http://www.w3.o <series confidence="0.95119114462180 <fictional_character confidence="0.9 wordnetid="109587565"> <research_worker confidence="0.95119 <header> <title>Hercule Poirot</title> <id>1000</id> <revision> <id>243685651</id> <timestamp>2008-10-07T16:45:26Z</time <contributor> <username>Lightmouse</username> <id>4469495</id> </contributor> </revision> <categories> <category>Hercule Poirot characters</ <category>Fictional private investiga <category>Hercule Poirot</category>

Format des documents

- Texte simple
- Postscript, PDF, MS Word, etc.
- HTML, XML, etc.
- Filtre de conversion. . .
- . . . ou lecture adaptée

❸Niveau lexical (1/3)

- Jeu de caractères
- Lexèmes vs. n-grammes
- Les lexèmes
  - Accents (`résumé` vs. `resume`)
  - Apostrophe (`Finland's capital`, `L'ensemble`)
  - Point (`O.N.U.`)
  - Tiret (`co-education`, `Hewlett-Packard`, `the hold-him-back-and-drag maneuver`)
  - Espace (`San Francisco`, `Max Os X`, `MacOs X`, `MacOsX`)
  - Tiret-Espace (`MSDOS`, `MS-DOS`, `MS DOS`, `San Francisco-Los Angeles`)
  - Autres (`C++`, `C#`, `M*A*S*H`, `mbeig@emse.fr`, etc.)

❸Niveau lexical (2/3)

- Les lexèmes (suite)
  - Chiffres, nombres, date
    - `3/12/91`
    - `Mar. 12, 1991`
    - `55 B.C.`
    - `B-52`
    - `My PGP key is 324a3de234cb23f`
    - `8G43560786151`
    - `100.2.86.144`

❸Niveau lexical (3/3)

- Des lexèmes aux termes d'indexation
  - Normalisation (casse, accents)
  - Liste de mots vides (mono-lingue, multi-lingue)
  - Lemmatisation (mono-lingue, multi-lingue)
  - Indexation contrôlée
- Reconnaissance de la langue

❸$_0$ Structure$_1$ des$_2$ documents$_3$

- Texte$_4$ comme$_5$ séquence$_6$ (position$_7$ des$_8$ occurrences$_9$ des$_{10}$ termes$_{11}$ dans$_{12}$ l$_{13}$'index$_{14}$)
- Champs$_{15}$-zone$_{16}$ (auteur$_{17}$-titre$_{18}$ etc$_{19}$. ; Expéditeur$_{20}$-Destinataires$_{21}$-Sujet$_{22}$ etc$_{23}$.)
- Structure$_{24}$ hiérarchique$_{25}$ (sections$_{26}$-titres$_{27}$, XML$_{28}$)
- Structure$_{29}$ inter$_{30}$-documents$_{31}$ (hypertexte$_{32}$)

❹Exemples de besoin d'information

TREC8 (1999)/WT10G

```
<top>
<num> Number: 401
<title> foreign minorities, Germany
<desc> Description:
What language and cultural differences impede the integration of foreign minorities in Germany?
<narr> Narrative:
A relevant document will focus on the causes of the lack of integration in a significant way;
that is, the mere mention of immigration difficulties is not relevant.  Documents that discuss
immigration problems unrelated to Germany are also not relevant.
</top>
```

INEX 2008/Wikipedia

```
<topic id="544" ct_no="6">
  <title>meaning of life</title>
  <castitle>//article[about(., philosophy)]//section[about(., meaning of life)]</castitle>
  <description>What is the meaning of life?</description>
  <narrative>I got bored of my life and started wondering what the meaning of life is. An
  element is relevant if it discusses the meaning of life from different perspectives, as
  long as it is serious. For example, Socrates discussing meaning of life is relevant, but
  something like "42" from H2G2 or "the meaning of life is cheese" from a comedy is
  irrelevant.  An element must be self contained. An element that is a list of links is
  considered irrelevant because it is not self-contained in the sense that I don't know
  in which context the links are given.</narrative>
</topic>
```

❺Langage de requêtes

- Lié au modèle de correspondance
- Requêtes booléennes

  INEX 2007/415: `space history astronaut cosmonaut engineer`
  INEX 2007/415: `space & history & (astronaut | cosmonaut | engineer)`
- Ensemble de mots (Ordre et répétition sans conséquence)
- Sac de mots (Répétition prise en compte)
- Liste de mots (Ordre des mots pris en compte)
- Opérateurs '+' '–' ""

  INEX 2007/420: `Shading Models "Phong Shading"`
  INEX 2008/550: `dna testing -forensic -maternity -paternity`
- Limite sur le nombre de mots
- Jokers (caractères d'expansion)
- Pondération ($e.g.$ Inquery)

  Doc. Indri:
  ```
  #combine( #wsum( 5.0 bbc.(title) 3.0 bbc.(anchor) 1.0 bbc)
            #wsum( 5.0 news.(title) 3.0 news.(anchor) 1.0 news ))
  ```
- Les champs-zones

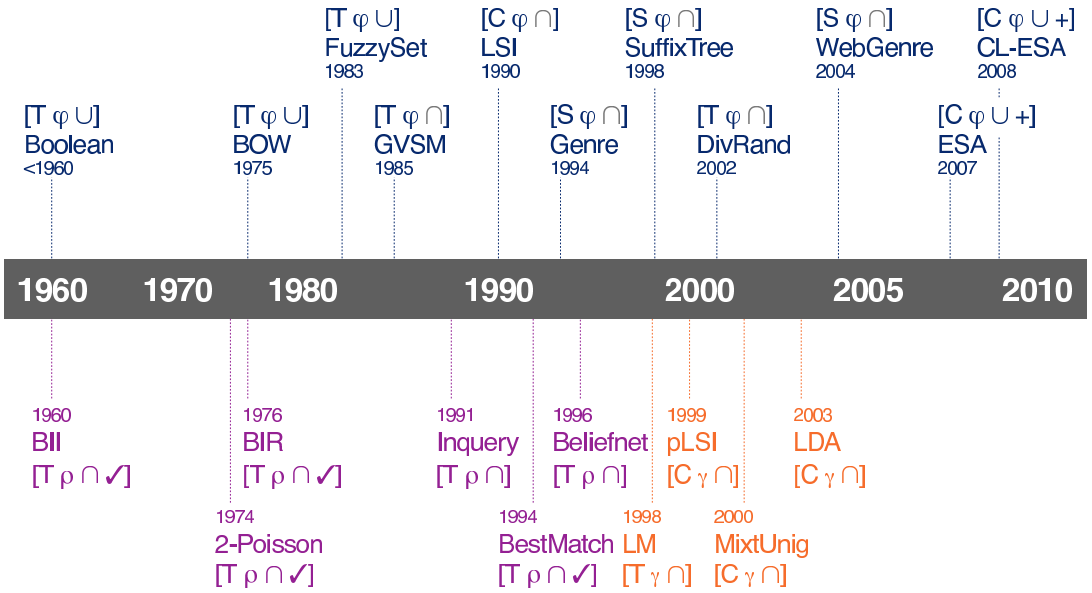
  Doc. Sphinx: `"hello world" @title "example program"~5 @body python -(php|perl) @* code`
- Correction orthographique, expansion ≪ sémantique ≫

Modèle de correspondance

Extrait de la présentation de Stein et al. à TIR 09

`http://www.uni-weimar.de/medien/webis/research/workshopseries/tir-09/talks/stein09-talk-collection-relative-representations-a-unifying-view-to-retrieval-models.pdf`

Retrieval Models



- Empirical models
- Probabilistic models
- Language models

Feature space	[T] terms [C] concepts [S] special
RSV foundation	[φ] sim. [ρ] relevance [γ] generation
Collection	[∪] open [∩] closed
Ext. knowledge	[✓] user feedback [+] collection

12 Stein@TIR [^] 31.08.09

Extraits de ❼ *run* et de ❽ jugements de pertinence

Run Terrier BM25	INEX 2008 qrels
544 Q0 272436 0 6.023203661344247 BM25b1.0	544 Q0 20347 26294 26299 1 1:26294
544 Q0 431221 1 5.929980409237743 BM25b1.0	544 Q0 261763 0 2258
544 Q0 2041642 2 5.929980409237743 BM25b1.0	544 Q0 326920 0 7124
544 Q0 85677 3 5.851558247760837 BM25b1.0	544 Q0 682628 0 2055
544 Q0 261763 4 5.8412106049374595 BM25b1.0	544 Q0 177316 572 4798 1915 1915:299 3711:273
544 Q0 2293590 5 5.790298320602847 BM25b1.0	544 Q0 1831754 6487 6490 1 1:6487
544 Q0 1137622 6 5.775183231207504 BM25b1.0	544 Q0 148681 0 9338
544 Q0 1564714 7 5.762694287565935 BM25b1.0	544 Q0 551722 326 7739 15 15:326
544 Q0 648543 8 5.674528105081118 BM25b1.0	544 Q0 2952344 0 1902
544 Q0 20347 9 5.670369214560328 BM25b1.0	544 Q0 233013 3796 3801 1 1:3796
544 Q0 682628 10 5.644216519443381 BM25b1.0	544 Q0 2293590 0 1061
544 Q0 2383086 11 5.62582225898192 BM25b1.0	544 Q0 238114 0 2095
544 Q0 3162419 12 5.621504108577819 BM25b1.0	544 Q0 191379 0 27994
544 Q0 309238 13 5.604499013011449 BM25b1.0	544 Q0 648543 1139 1666 423 423:1139
544 Q0 2728573 14 5.581378099146111 BM25b1.0	544 Q0 8753 0 21215
544 Q0 1600053 15 5.557569670375079 BM25b1.0	544 Q0 985414 255 2810 922 922:255
544 Q0 1001962 16 5.546589682639702 BM25b1.0	544 Q0 522975 0 7264
544 Q0 238114 17 5.539158393968342 BM25b1.0	544 Q0 986258 186 2921 340 340:186
544 Q0 3203352 18 5.534398492848405 BM25b1.0	544 Q0 2784 0 4178
544 Q0 2092783 19 5.534398492848405 BM25b1.0	544 Q0 1392796 0 2463
544 Q0 779206 20 5.529371216833702 BM25b1.0	544 Q0 1255122 0 7557
544 Q0 1128197 21 5.511419728294348 BM25b1.0	544 Q0 726508 0 11580
544 Q0 2082424 22 5.511419728294348 BM25b1.0	544 Q0 376862 969 5503 3431 3431:969

trec_eval (1/2)

trec_eval est le logiciel pour faire les évaluations RI. Plein de versions. La plus récente est la 9.0. On lui donne le fichier des jugements ($qrel$) puis le fichier des résultats retrouvés (run).

run

544	Q0	272436	0	6.023203661344247	BM25b1.0
qid	*iter*	*docno*	*rank*	*sim*	*run_id*
string	*string*	*string*		*float*	*string*

qrel

544	Q0	20347	1
qid	*iter*	*docno*	*rel*
string	*string*	*string*	*int*

```
rel, a non-negative integer less than 128, or -1 (unjudged)
```

trec_eval (2/2)

trec_eval produit **135** mesures, dont

- *Interpolated Recall - Precision Averages*
  Courbe de précision-rappel

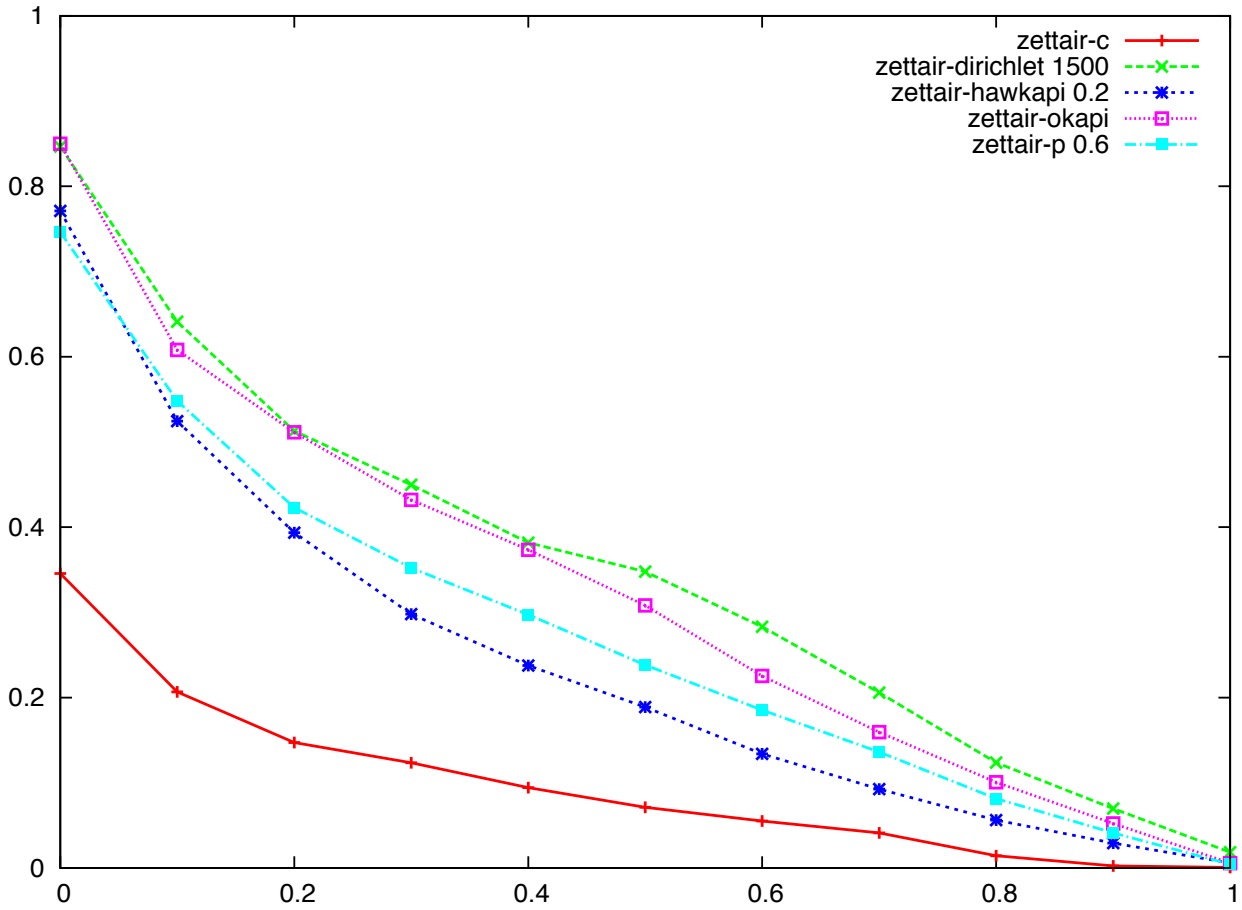
```
ircl_prn.0.00   all     0.8462
ircl_prn.0.10   all     0.6433
ircl_prn.0.20   all     0.5230
ircl_prn.0.30   all     0.4448
ircl_prn.0.40   all     0.3790
ircl_prn.0.50   all     0.3178
ircl_prn.0.60   all     0.2436
ircl_prn.0.70   all     0.1754
ircl_prn.0.80   all     0.1211
ircl_prn.0.90   all     0.0622
ircl_prn.1.00   all     0.0158
```

- *Mean Average precision*

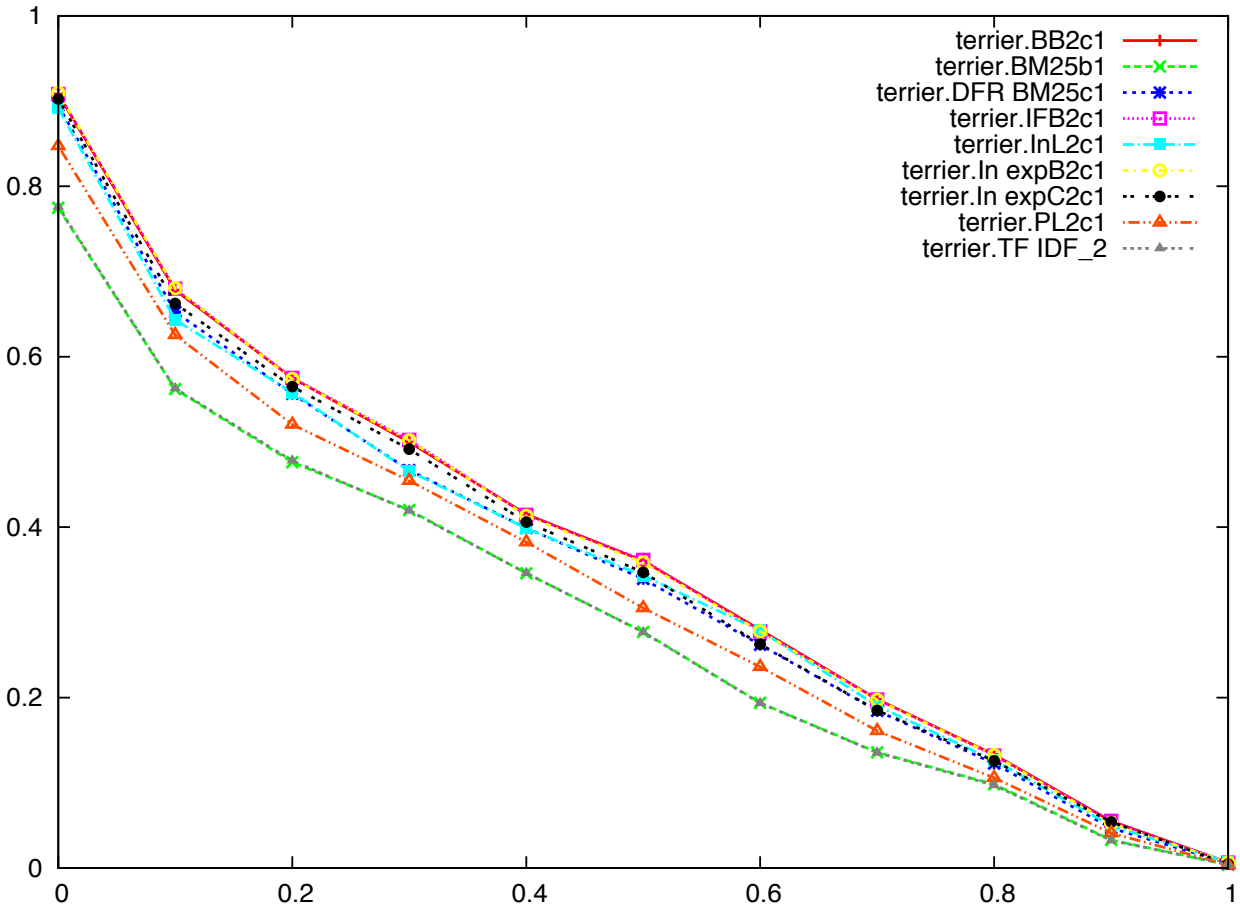
```
map             all     0.3204
gm_ap           all     0.2512
```

- *Precision at 5, 10, ... 1000 docs*
- *R-Precision*
- *Binary Preference*
- etc.

zettair : Courbes P/R (trec_eval et gnuplot)



terrier : Courbes P/R (trec_eval et gnuplot)



Sortie des résultats

Liste de résultats

- *snippets*
- accès au document
- tri
- facettes

Interface

- Interface graphique
- Entrée des requêtes
- Mode serveur
- Mode TREC

License

Langage de programmation

Stockage de l'index

Résumé des critères

1. Langage de programmation (C, C++, Java, perl, etc.)
2. License
3. Usage : Site, Poste de travail, Bibliothèque, évaluation ad'Hoc, Divers (Traitement de documents)
4. Collection : smart, TREC, par répertoire, par URL, incrémental
5. Format des documents : texte, ps, pdf, doc, HTML, XML
6. Jeu de caractères : ASCII (7 bits), ISO–8859 (8 bits), Unicode (UTF-8, UTF-16, etc.)
7. Indexation : lexèmes, n-grammes, nombres, dates, sigles, adresses couriel, url, etc.
8. Normalisation des lexèmes : casse, mots vides, lemmatisation, vocabulaire contrôlé

9. Multi-lingue
10. Indexation : positions des termes, champs, structure hiérarchique, structure hypertextuelle
11. Langage de requêtes : booléen, opérateur $+$, opérateur '$-$', jokers, poids, etc.
12. Termes : appariement approché, correction orthographique, expansion $\ll$ sémantique $\gg$
13. Modèle d'appariement : vectoriel, BM25, modèle de langue, etc.
14. Résultats : $snippets$, documents entiers, tri/sélection, facettes
15. Interface : graphique, mode serveur, mode HTTP, mode TREC
16. Performances

Étude de Middleton et Baeza-Yates, 2007

A Comparison of Open Source Search Engines

29 outils listés, 12 testés. (0 : disparu, 1 trouvé, + commenté)

paralyzed	0	ASPSeek	Lucene app.	.	Nutch
paralyzed	0	BBDBot	+	+	Omega/Xapian
slow	+	Datapark	+	0	OmniFind IBM Yahoo ! Ed.
paralyzed	0	ebhath	slow	1	OpenFTS
paralyzed	0	Eureka	paralyzed	0	PLWeb
+	1	ht://Dig	+	1	SWISH-E
+	+	Indri/Lemur	+	1	SWISH++
paralyzed	1	ISearch	+	+	Terrier
+	0	IXE	paralyzed	0	WAIS/freeWAIS
+	+	Lucene/Nutch, Solr	slow	1	WebGlimpse
paralyzed	+	Managing Gigabytes (MG)	XML	1	XML Query Engine
+	1	MG4J	+	0	XMLSearch
slow	1	mnoGoSearch	XML	+	Zebra
paralyzed	0	MPS Information Server	+	+	Zettair
slow	1	Namazu	**Ajouté : smart, Cheshire3, Sphinx, Wumpus**		

Les liens, les versions (1/2)

Classés par date de dernière version

smart	1992	11.0	C	?	ftp : //ftp.cs.cornell.edu/pub/smart/
mg	1999/08	1.2.1	C	GPL	http : //ww2.cs.mu.oz.au/mg/
		1.3g	C	GPL	http : //www.nzdl.org/html/mg.html
bow	2002/02	20020213	C	GPL	http : //www.cs.cmu.edu/ mccallum/bow/
ht://Dig	2004/06	3.2.0b6	C C++ perl	LGPL	http : //www.htdig.org/
Swish++	2006/05	6.1.5	C	GPL	http : //swishplusplus.sourceforge.net/
zettair	2006/09	0.9.3	C	GPL	http : //www.seg.rmit.edu.au/zettair/
clairlib	2009/09	1.08	C		http : //www.clairlib.org/
Swish-e	2009/04	2.4.70	C perl	GPL	http : //www.swish − e.org/
XQEngine	2009/07	0.63	Java	?	http : //sourceforge.net/projects/xqengine/
OpenFTS	2009/12	0.40	C perl	GPL	http : //openfts.sourceforge.net/
iSearch	2010/06	2.24	php	L	http : //www.isearchthenet.com/isearch/
dpSearch	2011/03	4.53	C perl BdD	GPL	http : //www.dataparksearch.org/
namazu	2011/07	2.0.21	C perl	GPL	http : //www.namazu.org/index.html.en

Les liens, les versions (2/2)

wumpus	2011/11	2011-11-10	C++	GPL	http://www.wumpus-search.org/
WebGlimpse	2012/09	2.21.0	perl	L	http://webglimpse.net/
Glimpse	2012/09	4.18.6	C	L	http://webglimpse.net/
mg4j	2013/02	5.2.1	Java	GPL	http://mg4j.di.unimi.it/
mnoGoSearch	2013/12	3.3.15	C	GPL	http://www.mnogosearch.org/
zebra	2014/04	2.0.59	C	GPL	http://www.indexdata.com/zebra
terrier	2014/06	4.0	Java	Mozilla	http://terrier.org/
xapian	2014/06	1.2.18	C++	GPL	http://www.xapian.org/
indri	2014/07	5.7	C++	L	http://www.lemurproject.org/
cheshire	2014/07	1.1.8	Python	L	http://www.cheshire3.org/
lucene	2014/09	4.10.1	Java	Apache	http://lucene.apache.org/core/
solr	2014/09	4.10.1	Java	Apache	http://lucene.apache.org/solr/
sphinx	2014/10	2.2.5	C++	GPL	http://sphinxsearch.com/

Usage

- Bibliothèques (API) pour construire une application
  lucene xapian (terrier) clairlib bow
- Recherche sur un site
  sphinx dpSearch xapian/omega
  lucene/solr namazu mnoGoSearch
  ht://Dig Swish-e Swish++
- Recherche sur poste de travail
  wumpus lucene/ ? xapian/ ?
- Recherche dans des catalogues de bibliothèques
  cheshire zebra mg
- Evaluation RI ad'hoc
  smart mg zettair wumpus terrier lucene indri

Le vétéran : smart

Expérimental. Classification supervisée et non supervisée, $ad'hoc$, TREC, expansions de requêtes, retour de pertinence.

- 70 000 lignes de C

☺ organisation flexible

- vectoriel, multiples pondérations (on peut en ajouter)
- représentation explicite des vecteurs des documents

☺ diversités des tâches

☹ documentation

☹☹☹ configuration

☹ limitations (taille, etc.)

☹ pas de positions des termes, . . .

Managing gigabytes : mg

Expérimental. *Ad'hoc*, TREC.

- 50 000 lignes de C
- modèle booléen et vectoriel basique (`nts.ntn`)

☺ compression des documents, des index

☺ modèle booléen et vectoriel

☺ efficient

☹ difficile d'accéder aux index pour implanter d'autres méthodes de recherche

☹ code difficile à lire (macros)

☹ documentation du code insuffisante

☹ pas de positions des termes, pas de vecteurs directs (possible à ajouter)

Expérimental. *Ad'hoc*, TREC.

- 68 000 lignes de C (lucy : 20 000)
- Okapi (k1, k3, b), pivoted-cosine, cosine, hawkapi (alpha), LM Dirichlet (mu)

☺ très efficient, assez facile à lire (plus facile dans la version plus ancienne lucy)

☺ très facile d'ajouter des correspondances basées sur tf, idf, longueurs des docs.

☺ Lexicalisation : `O.N.U.`, balises alla XML

☺ Position des occurrences dans les index.

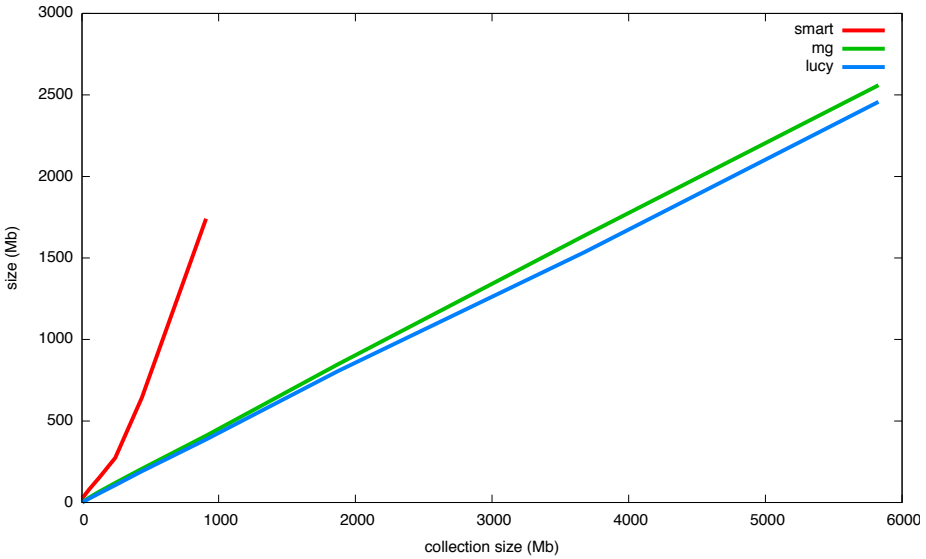
☺ Requêtes avec expressions

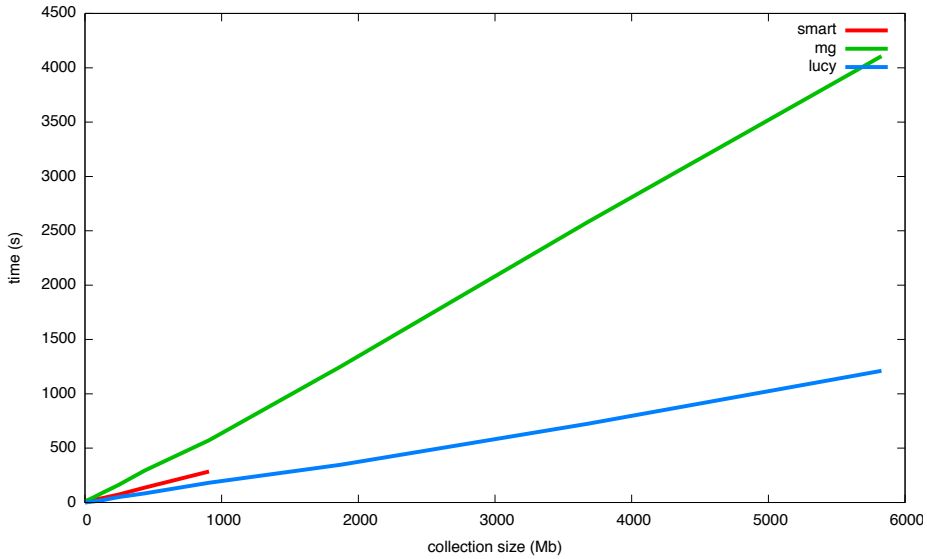
☹ Pas de vecteurs des documents.

☹ Pas de lecture XML complète

smart mg zettair/lucy

Taille des index et temps d'indexation





bow

Expérimental. Classification supervisée et non supervisée, $ad'hoc$, TREC.

- 47 000 lignes de C
- $tf \cdot idf$ (variations sur tf, idf et icf, possible d'en ajouter)

☺ code d'une remarquable clarté

☺ n-grammes

☺ mode serveur

Wumpus

Expérimental. *ad'hoc*, TREC, *Desktop search*.

- 68 000 lignes de C++
- Okapi, QAP, or QAP2 (avec contraintes booléennes)
- ☺ Support des GCL (Cf. Clarke et al. 1995)
- ☺ Support de XPath
- ☺ Mode serveur
- ☺ Lemmatisation à l'interrogation, jokers sur préfixes
- ☺ Indexation dynamique

C.L.A. Clarke, G.V. Cormack, and F.J. Burkowski. An Algebra for Structured Text Search and a Framework for its Implementation. The Computer Journal, 38(1):43-56, 1995.

Bibliothèque en C++, *Site search*, *Web search*, développement de solutions de recherche.

- 160 000 lignes de C++
- BM25 et filtres booléens
- Champs-zones
- Requêtes : AND, OR, NOT, XOR, $+/-$, NEAR, ADJ, "", avec ou sans lemmatisation, jokers, synonymes, intervalles (dates, nombres)
- correction orthographique
- mode serveur
- **Omega** : une application fournie, pour du *Site search*

☺ documentation

Sphinx

Pragmatique. *Site search*

- 133 000 lignes de C++
- BM25 + proximité (*phrase search*)
- Sources : BdD, fichiers textes, HTML, BaL, etc.

☺ Scalable, parallélisable.

- Requêtes : booléen, expressions, proximité, champs
- Zones (champs)
- Mots vides, lemmatisation
- Jeux de caractères divers (iso, UTF8)

dpSearch

Site search

- 130 000 lignes de C
- *Relevancy* (vectoriel), *Popularity rank*, *Neural Network*, date de dernière modification, combinaison de pertinence et de popularité.

☺ Dictionnaires de synonymes

☺ Lexicalisation (*tokenizing*) pour le chinois, le japonais, le coréen et le thaï.

- Java (Windows, Mac OS X, Linux and Unix), 121 000 lignes
- Open Source (Mozilla Public Licence).
- Desktop search (included app.), batch
  - Easily scriptable
  - Built-in evaluation tools
- Collection
  - Mulitlingual, UTF characters
  - common desktop file formats (HTML, PDF, .doc, .xls, .ppt)
  - TREC research collections
  - HTTP fetch
  - Support for changing the tokenisation, stopwords, stemmer
- Incremental indexing and retrieval capabilities
- Indexation : fields and word positions
- Query language

Advanced query language that supports synonyms, +/- operators, phrase and proximity search, and fields.

- Search models : (126) DFR, BM25, LM, TF-IDF
  - Query Expansion (pseudo-relevance feedback)
- Interface : desktop, command-line and Web based querying
- Misc. :
  - Indexing support for query-biased summarisation
  - Learning-to-rank support enables
- Scalability
  - at least 50 million documents
  - larger collections with Hadoop MapReduce distributed indexing scheme
- Extensibility : Modular and open indexing and querying APIs

- Lucene Core

  Java-based indexing and search technology, as well as spellchecking, hit highlighting and advanced analysis/tokenization capabilities.
- SolrTM

  high performance search server built using Lucene Core, with XML/HTTP and JSON/Python/Ruby APIs, hit highlighting, faceted search, caching, replication, and a web admin interface.
- Open Relevance Project

  subproject with the aim of collecting and distributing free materials for relevance testing and performance. (CLOSED 11/09/2014)
- PyLucene is a Python port of the Core project.

- Java, Cross-Platform Solution
- Powerful, Accurate and Efficient Search Algorithms
  - ranked searching
  - pluggable ranking models, including the Vector Space Model and Okapi BM25
- Query language
  - many query types : phrase queries, wildcard queries, proximity queries, range queries and more
  - fielded searching (e.g. title, author, contents)
  - multiple-index searching with merged results
- allows simultaneous update and searching
- flexible faceting, highlighting, joins and result grouping
- fast, memory-efficient and typo-tolerant suggesters configurable storage engine (codecs)

- Scalability
  - over 150GB/hour on modern hardware
  - small RAM requirements – only 1MB heap
  - incremental indexing as fast as batch indexing
  - index size roughly 20-30% the size of text indexed

Indri

Expérimental. *ad'hoc*, TREC.

- 126 000 lignes de C++

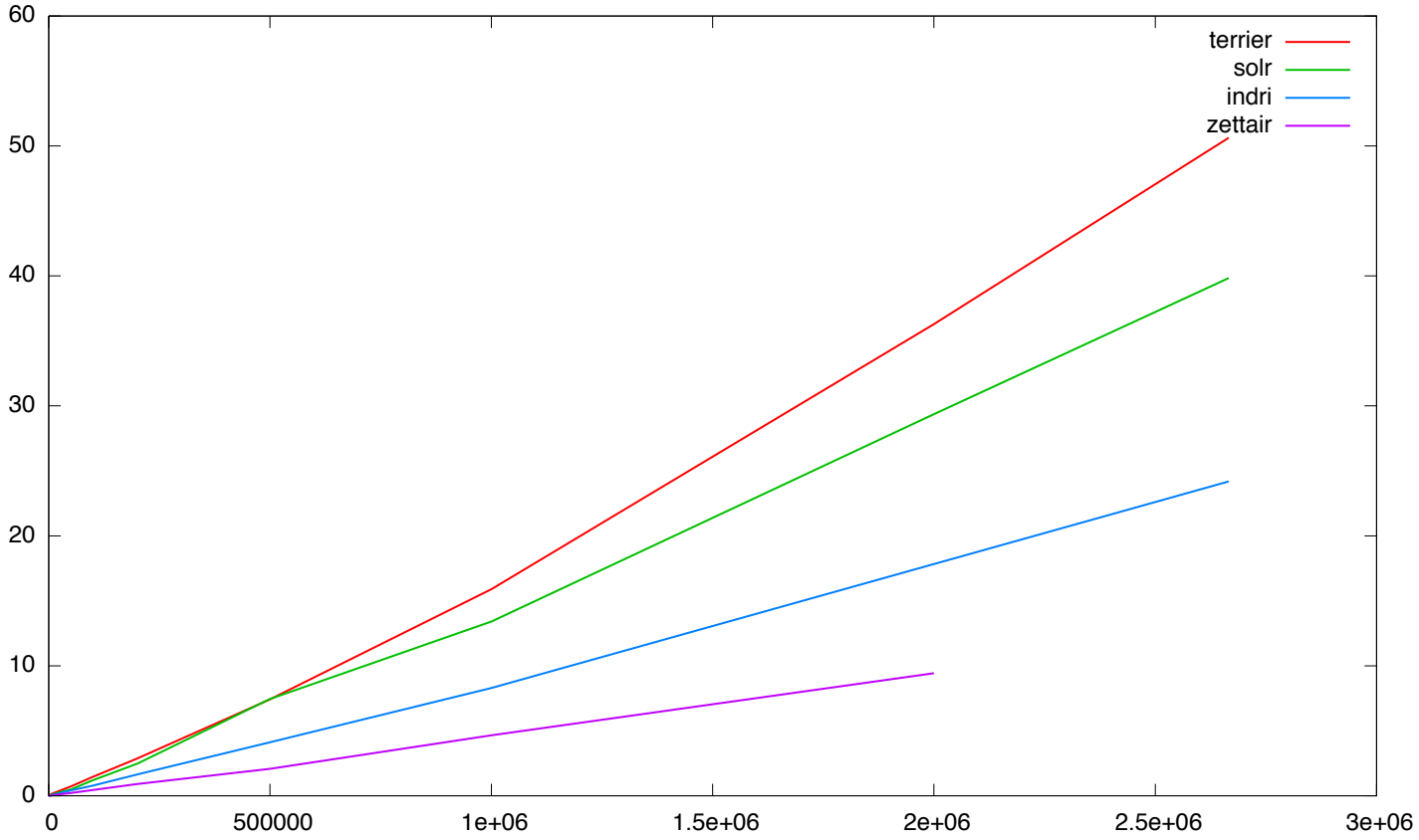
– C++ (Windows, Linux, Solaris and Mac OS X)
– Usage

  API can be used from Java, PHP, or C++
– Collection
  – Language independent tokenization of UTF-8 encoded documents.
  – Parses PDF, HTML, XML, and TREC documents
  – Word and PowerPoint (Windows only)
– Indexation
  – Field retrieval
  – Passage retrieval
– Query language

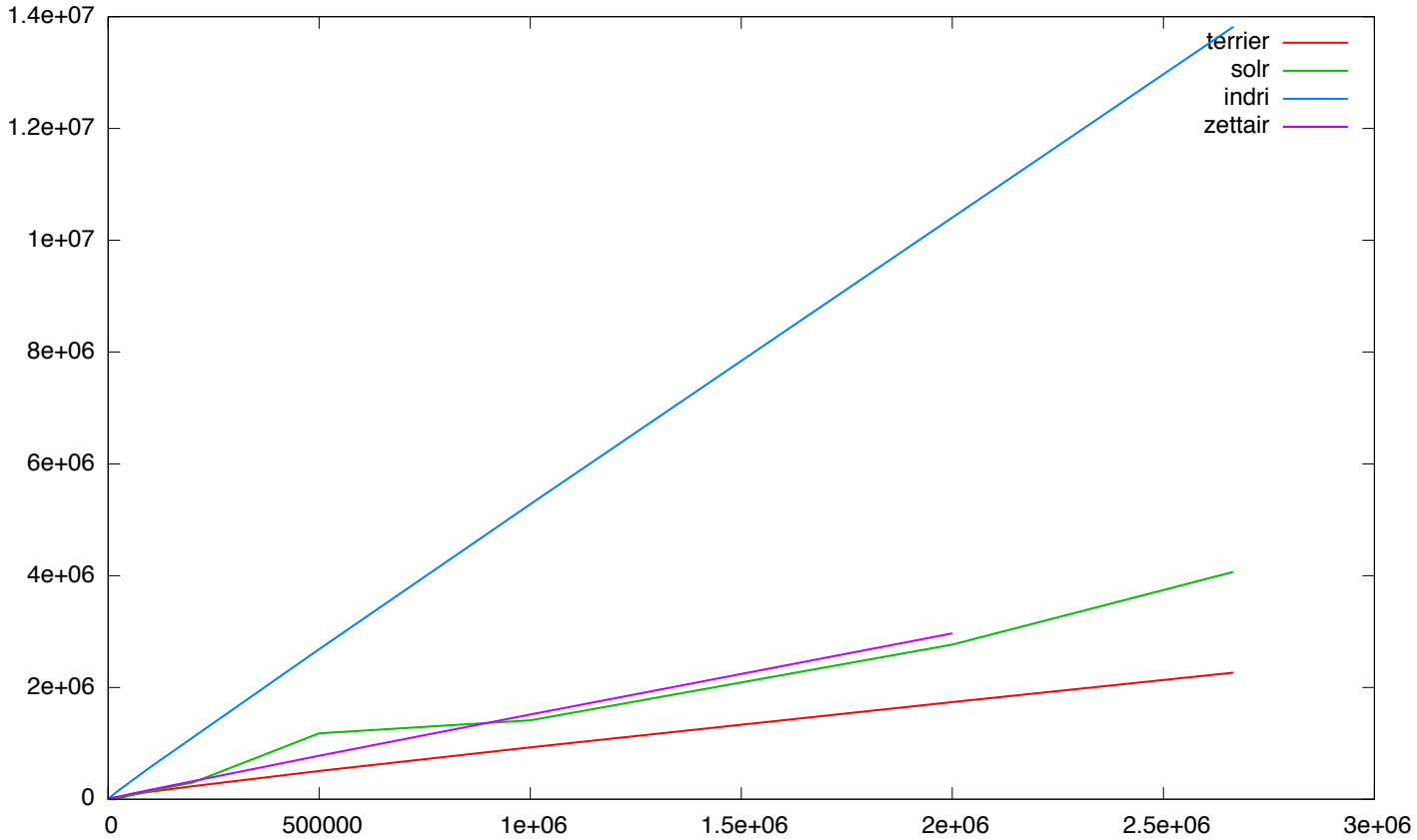
  - Supports popular structured query operators from INQUERY `#weight #combine #or #not #wand #wsum #max` etc.
  - Suffix-based wildcard term matching
- Interface
  - command line tools
  - Java user interface
- Scalability
  - Can be used on a cluster of machines for faster indexing and retrieval
  - Scales to terabyte-sized collections

indri solr terrier zettair : temps d'indexation

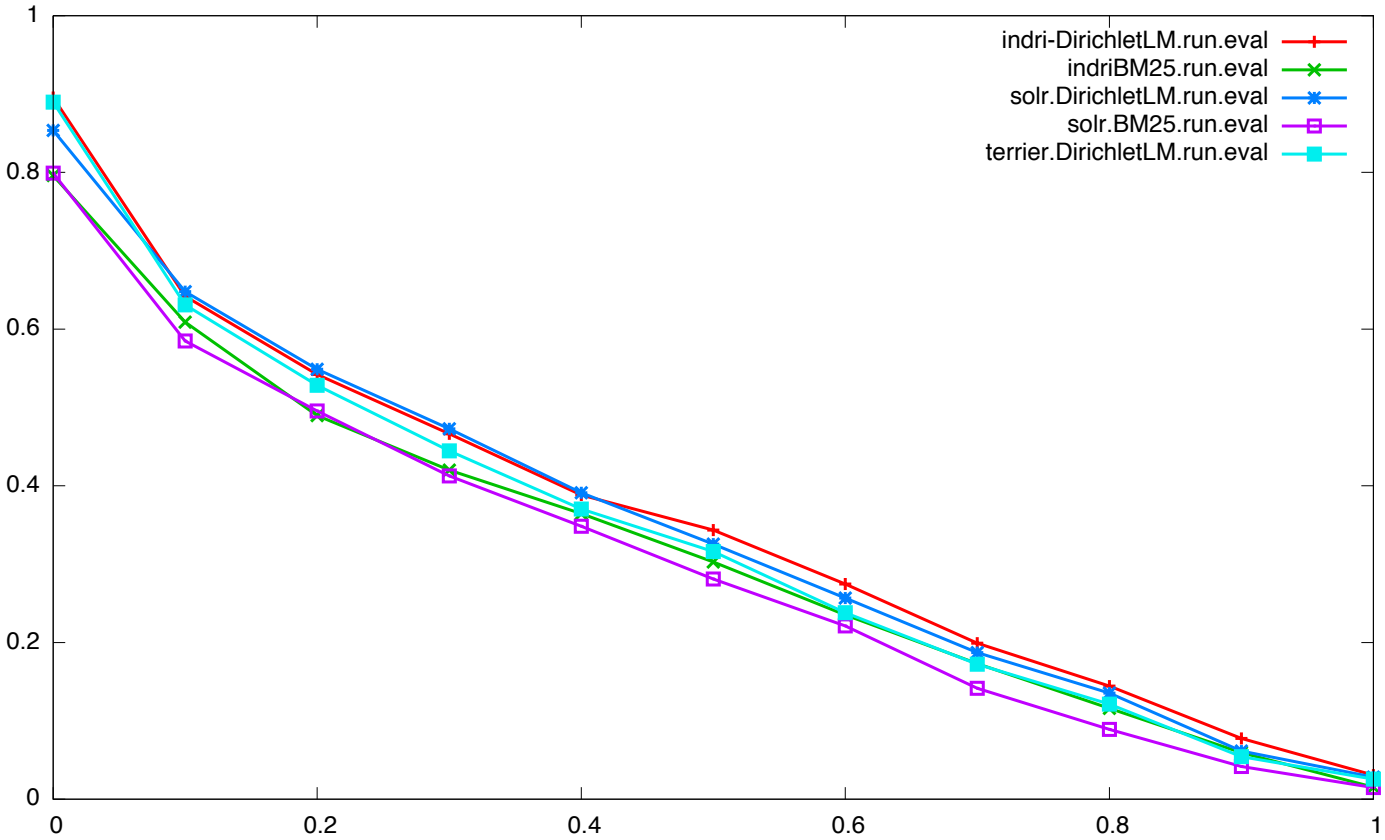
En minutes. En abscisse : le nombre de documents. Collection de INEX Wikipedia 2008, 2 666 190 documents convertis en ASCII, sans balises.



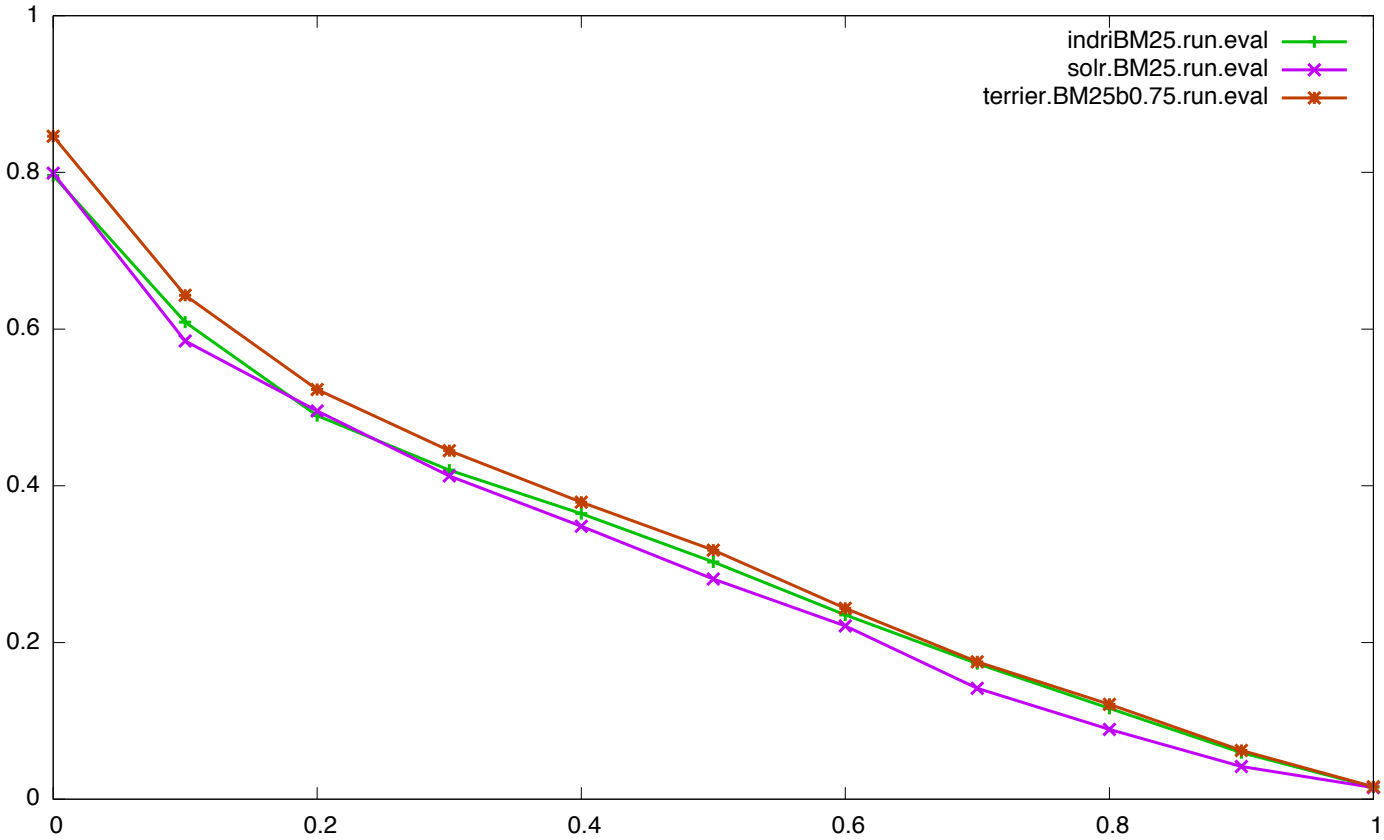
indri solr terrier zettair : taille d'index



indri solr terrier : rappel-précision



indri solr terrier : rappel-précision BM25



indri solr terrier : rappel-précision LM Dirichlet