Modern Information Retrieval

Appendix A

Open Source Search Engines

with Christian Middleton

Introduction
Search Engines
Comparison Methodology
Experimental Results

Introduction

- There are many reasons to use an **open search engine** in a Web site or other IR applications inside a company
  - cost considerations
  - commercial engine has focus on larger sites
  - specific needs that imply code customization
- For small to medium traffic Web sites is an interesting alternative
  - no licensing fees
  - source code available, so customization is possible
  - but maintenance and performance might be an issue

Introduction

- Open source search engines might be classified by
  - programming language of implementation
  - index data structure
  - search capabilities: Boolean, fuzzy, stemming
  - ranking function
  - files they can index: HTML, PDF, Word, plain text
  - online and incremental indexing
  - maintenance activity and people needed
- For adopting a search engine, one need to understand performance
  - behavior under distinct load conditions
  - degradation as load increases

Open Source Search Engines

Search Engine	Update	Version	Observation
ASPSeek	2002	N/A	Project is paralyzed.
BBDBot	2002	N/A	Last update was on 2002.
Datapark	13/03/2006	4.38	
ebhath	N/A	N/A	No existing website.
Eureka	N/A	N/A	Website is not working.
HtDig (ht://Dig)	16/06/2004	3.2.0b6	
Indri	22/06/2009	2.10	
ISearch	02/11/2000	1.75	Software not actively maintained.
Lucene	05/11/2009	2.9.1	
Managing Gigabytes (MG)	01/08/1999	1.2.1	
MG4J	06/06/2009	3.0	
mnoGoSearch	29/10/2009	3.3.9	
MPS Inform. Server	01/09/2000	6.0	
Namazu	23/09/2009	2.0.20	

Open Source Search Engines

Search Engine	Update	Version	Observation
Nutch	23/03/2009	1.0	Subproject of the Lucene project.
Omega	08/04/2006	0.9.5	Based on Xapian library.
OmniFind IBM Yahoo!	2009	8.4.2	
OpenFTS	05/04/2005	0.39	
PLWeb	16/03/1999	3.0.4	Code no longer available.
SWISH-E	04/04/2009	2.4.7	
SWISH++	25/01/2008	6.1.5	
Terrier	29/01/2009	2.2.1	
WAIS & freeWAIS	N/A	N/A	Software is outdated.
WebGlimpse	19/12/2008	4.18.6	Uses Glimpse as the indexer.
XML Query Engine	02/04/2005	0.69	XML search engine.
Zebra	05/11/2009	2.0.42	XML search engine.
Zettair	09/2006	0.9.3	

27 open source engines considered in 2009

Preliminary Selection of Engines

- Project outdated, not maintained, paralyzed
  1. ASPSeek
  2. BBDBot
  3. ebhath
  4. Eureka
  5. ISearch
  6. MPS Information Server
  7. PLWeb
  8. WAIS/freeWAIS
  - **19 engines left for consideration**
- Eliminate engines that depend on other or have a special purpose
  9. Managing Gigabytes (MG)
  10. Nutch
  11. XML Query Engine
  12. Zebra
  - **15 engines remain for consideration**

Preliminary Selection of Engines

- Preliminary indexing tests showed 5 very slow engines
  13. Datapark
  14. mnoGoSearch
  15. Namazu
  16. OpenFTS
  17. Glimpse
  - **10 engines left for consideration**
- 10 engines selected for experimental comparison
  1. HtDig
  2. Indri
  3. Lucene
  4. MG4J
  5. Omega
  6. OmniFind
  7. SWISH-E
  8. SWISH++
  9. Terrier
  10. Zettair

The Ten Engines Selected

Search Engine	Storage[f]	Increm. Index	Results Excerpt	Results Template	Stop words	File types[e]	Stemming	Fuzzy Search	Sort[d]	Ranking	Search Type[c]	Indexer Lang.[b]	License[a]
HtDig	1	■	■	■	■	1,2	■	■	1	■	2	1,2	4
Indri	1	■	■	■	■	1,2,3,4	■	■	1,2	■	1,2,3	2	3
Lucene	1	■	□	□	■	1,2,4	■	■	1	■	1,2,3	3	1
MG4J	1	■	■	■	■	1,2	■	□	1	■	1,2,3	3	6
Omega	1	■	□	■	■	1,2,4,5	■	□	1	■	1,2,3	2	4
OmniFind	1	■	■	■	■	1,2,3,4,5	■	■	1	■	1,2,3	3	5
SWISH-E	1	■	□	□	■	1,2,3	■	■	1,2	■	1,2,3	1	4
SWISH++	1	■	□	□	■	1,2	■	□	1	■	1,2,3	2	4
Terrier	1	□	□	□	■	1,2,3,4,5	■	■	1	■	1,2,3	3	7
Zettair	1	■	■	□	■	1,2	■	□	1	■	1,2,3	1	2

10 Engines Selected

- Conventions for table in previous slide

(a) 1:Apache,2:BSD,3:CMU,4:GPL,5:IBM,6:LGPL,7:MPL,8:Comm,9:Free	
(b) 1:C, 2:C++, 3:Java, 4:Perl, 5:PHP, 6:Tcl	
(c) 1:phrase, 2:Boolean, 3:wild card.	
(d) 1:ranking, 2:date, 3:none.	■ Available
(e) 1:HTML, 2:plain text, 3:XML, 4:PDF, 5:PS.	□ Not Available
(f) 1:file, 2:database.	
(g) Commercial version only.	

Methodology

- Comparison tasks for 10 engines selected
  1. Obtain a document collection in HTML
  2. Determine a tool to use for monitoring the performance of the search engines
  3. Install and configure each of the search engines
  4. Index each document collection
  5. Process and analyze index results
  6. Perform a set of preselected searching tasks
  7. Process and analyze the search results

Document Collections

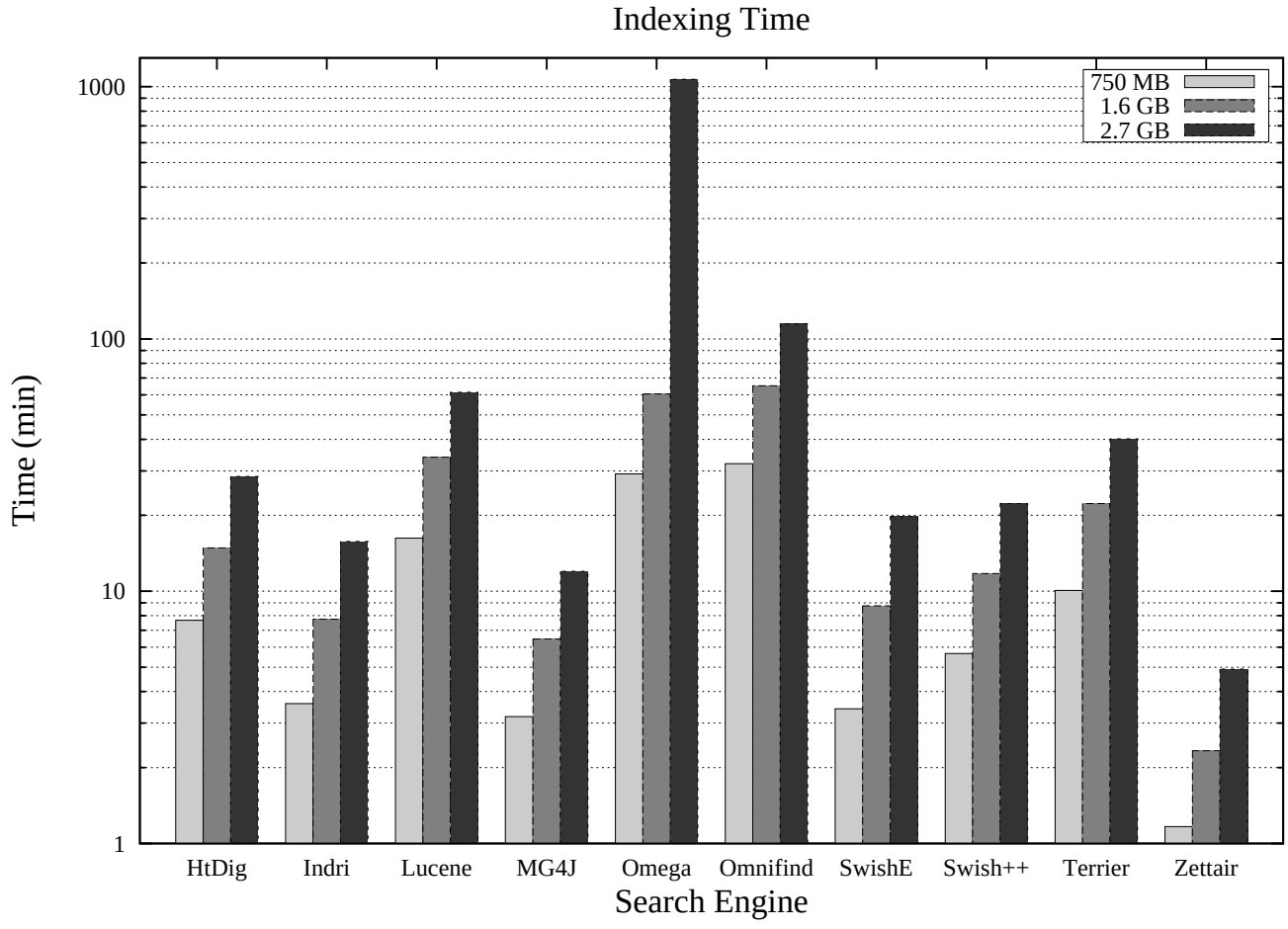
- Collections ranging from 1 GBytes to 10 GBytes
- **3 TREC-4 subcollections**
  - a first subcollection with 1,549 documents (750 MB)
  - a second subcollection with 3,193 documents (1.6 GB)
  - a third subcollection with 5,572 documents (2.7 GB)
- **4 WT10g subcollections**
  - a first subcollection occupying 2.4 GB
  - a second subcollection occupying 4.8 GB
  - a third subcollection occupying 7.2 GB
  - a fourth subcollection occupying 10.2 GB

Evaluation Tests

- 4 different evaluation tests
  - Test A – Indexing: index document collection with each search engine and record elapsed time and resource consumption
  - Test B – Incremental Indexing: time required to build incremental indexes.
  - Test C – Search Performance: query processing time of the engines, performance
  - Test D – Search Quality: quality of results produced by each engine, using precision-recall metrics
- Computer used for running tests
  - Pentium 4HT 3.2 GHz processor, 2.0 GB RAM, SATA hard disk driver, Debian Linux (Kernel 2.6.15)

Test A — Indexing

- Indexing of the 3 TREC-4 Subcollections



- **Omega and Omnifind performed poorly**

Test A — Memory and CPU

Size	750MB			1.6GB			2.7GB		
Search Engine	Max. CPU	Max. RAM	RAM Use	Max. CPU	Max. RAM	RAM Use	Max. CPU	Max. RAM	RAM Use
HtDig	100.0%	**6**.4%	C	100.0%	**6**.4%	C	88.9%	**6**.4%	C
Indri	100.0%	7.3%	L-S	97.5%	8.0%	L-S	88.6%	9.7%	L-S
Lucene	99.4%	20.0%	L	100.0%	38.3%	L	99.2%	59.4%	L
MG4J	100.0%	23.4%	C	100.0%	48.0%	C	100.0%	70.4%	C
Omega	100.0%	26.8%	L	99.2%	52.1%	L	94.0%	83.5%	L-C
OmniFind	78.4%	17.6%	S	**83.3**%	18.3%	S	83.8%	19.5%	S
Swish-E	100.0%	16.2%	L	98.9%	31.9%	L	98.8%	56.7%	L
Swish++	99.6%	24.8%	S	98.5%	34.3%	S	98.6%	54.3%	S
Terrier	99.5%	58.1%	S-C	99.4%	78.1%	S-C	98.7%	86.5%	S-C
Zettair	**77.2**%	20.2%	L	98.1%	22.3%	L	**82.7**%	23.1%	L
RAM behavior: C – constant, L – linear, S – step.									

- **All engines consumed close to 100% of CPU**

Test A — Memory and CPU

- 6 different patterns of memory consumption in previous slide
  - *constant* (C) – memory consumed remained constant;
  - *linear* (L) – memory consumed grew linearly with the index size;
  - *step* (S) – memory consumed grew initially, remained constant for a while, and resumed a pattern of growth afterwards;
  - *linear-step* (L-S) – a combination of linear growth with a step behavior;
  - *linear-constant* (L-C) – a combination of linear growth with a constant behavior; and
  - *step-constant* (S-C) – a combination of step behavior followed by constant memory consumption.

Test A — Memory and CPU

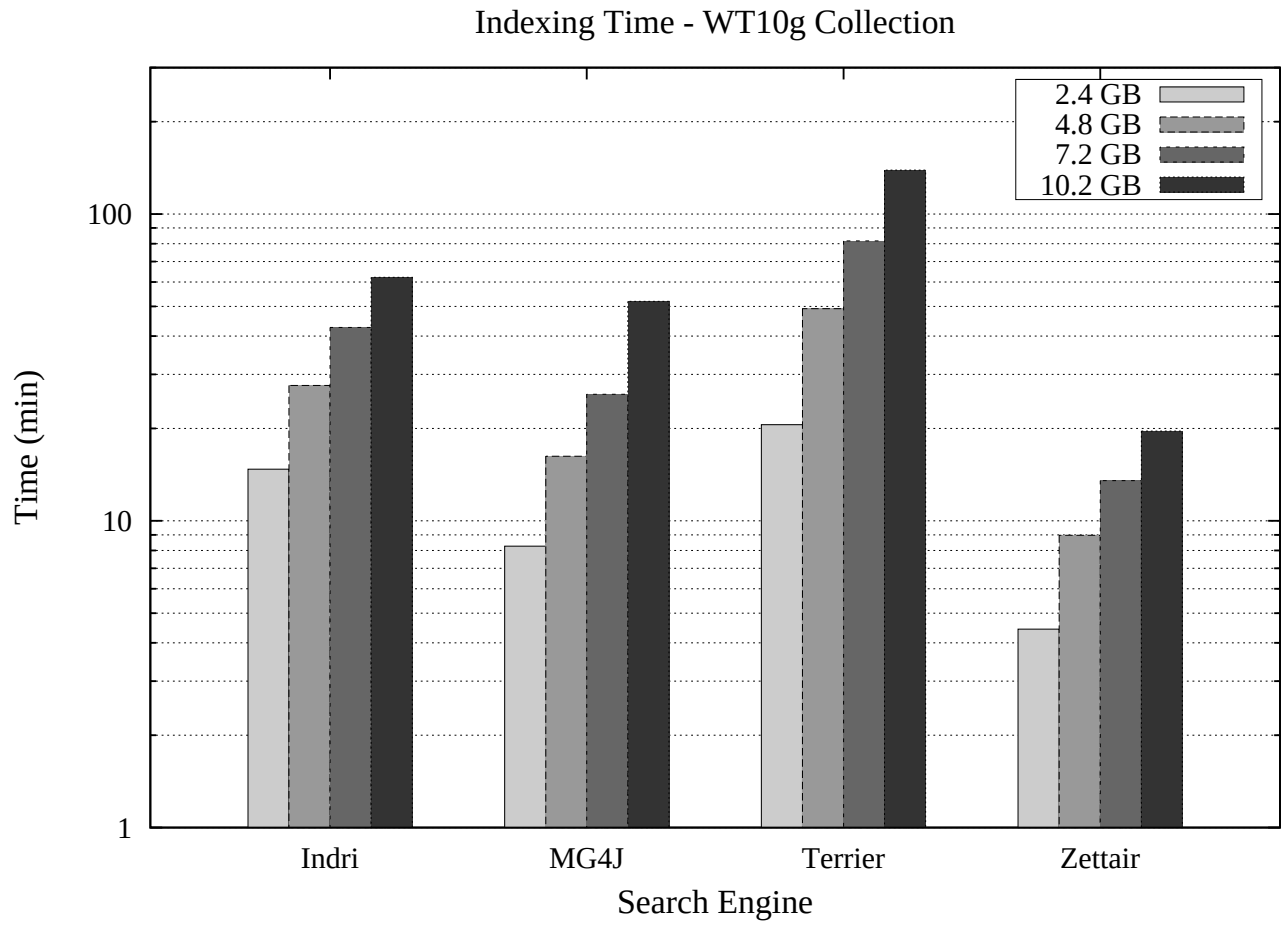
- Memory consumption pattern of the 10 engines
  - HtDig and MG4J: constant (C)
  - Lucene, Omega, Swish-E, and Zettair: linear growth (L)
  - Swish++ and OmniFind: step-like behavior (S)
  - Indri: linear growth, then decrease, afterwards linear (L-S)
  - Terrier: step-like growth, then constant (S-C)
  - Omega: linear growth, then constant (L-C)

Test A — Index Size

Search Engine	Index Size		
	750MB	1.6GB	2.7GB
HtDig	108%	92%	104%
Indri	61%	58%	63%
Lucene	**25**%	**23**%	**26**%
MG4J	30%	27%	30%
Omega	104%	95%	103%
OmniFind	175%	159%	171%
Swish-E	31%	28%	31%
Swish++	30%	26%	29%
Terrier	51%	47%	52%
Zettair	34%	31%	33%

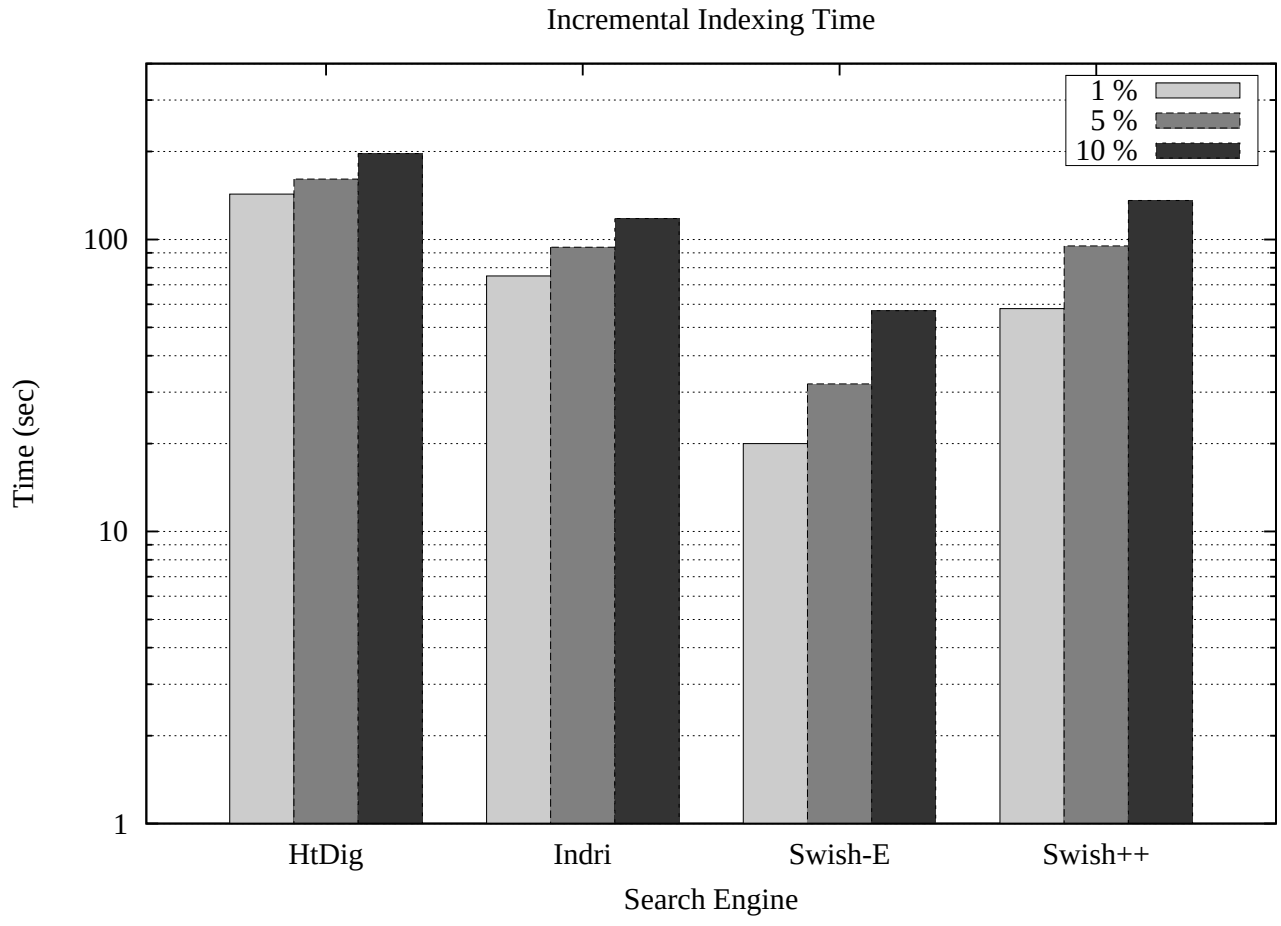
- **Best: Lucene, MG4J, Swish-E, Swish++, and Zettair: between 25%–35% of collection size**

Test A — Indexing WT10g



- **Indri, MG4J, Terrier, and Zettair: only engines to finish in linear time**

Test B — Incremental Indexing

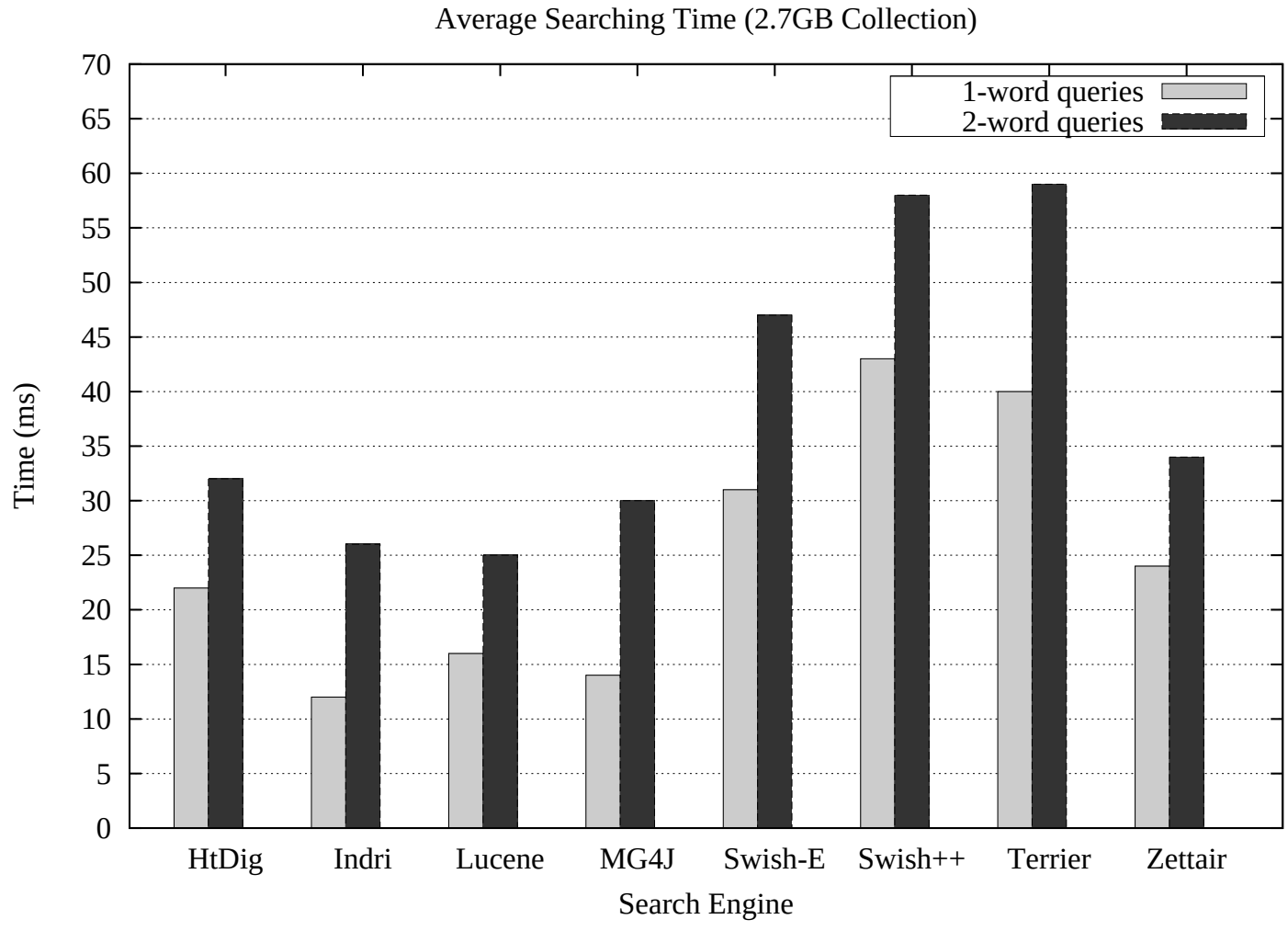


- Incremental indexing (1%, 5%, 10%) of 1.6GB collection
  - **Indri, MG4J, Terrier, Zettair: finished efficiently**

Test C — Search Performance

- We tested the 8 search engines that indexed efficiently
  - HtDig, Indri, Lucene, MG4J
  - Swish-E, Swish++, Terrier, Zettair
- To create the queries, we randomly selected 1 or 2 words using
  - original distribution of the words (power law)
  - uniform distribution over the 5% most frequent words (popular queries)
  - uniform distribution over the 30% least frequent words (rare queries)

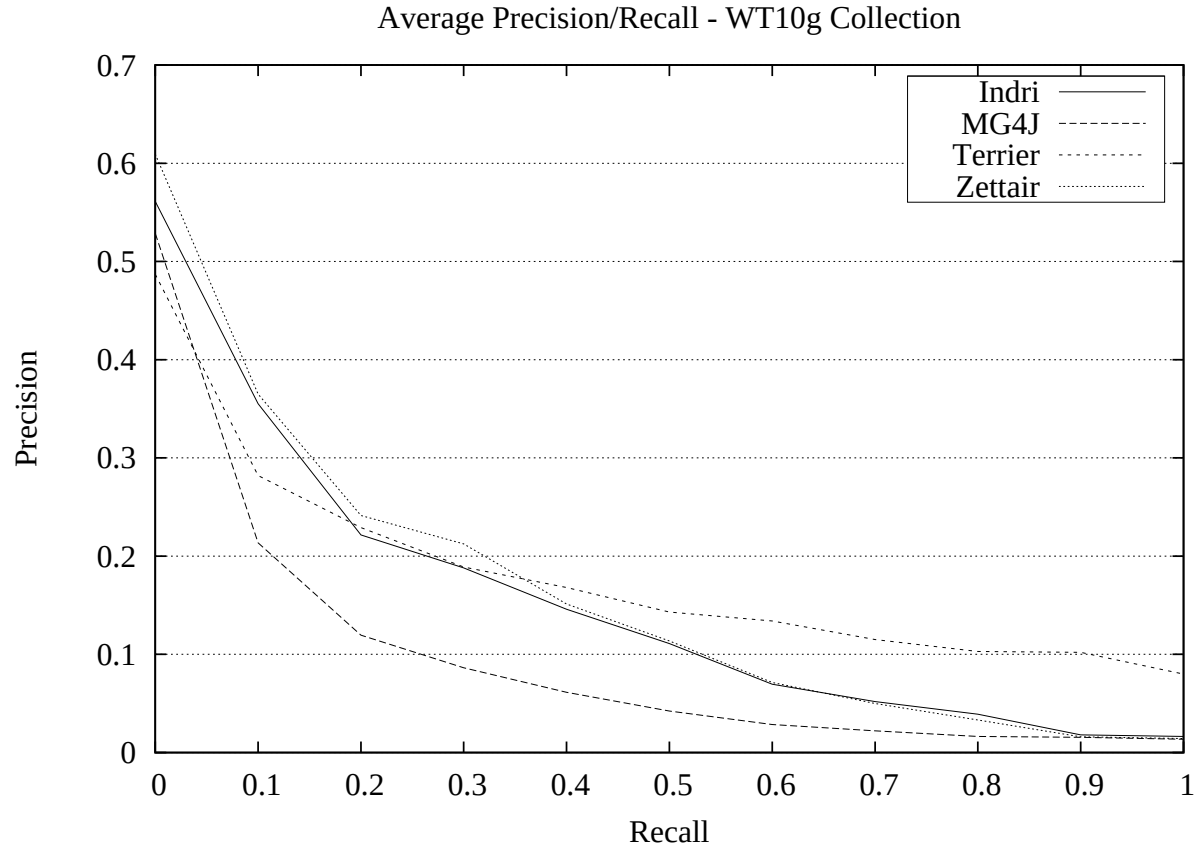
Test C — Search Performance



- **Indri and Lucene: fastest engines**

Test D — Search Quality

- WT10g collection used
  - 50 topic queries of the TREC-2001 Web track
  - interpolated precision at 11-pt recall levels



Test D — Search Quality

Search Engine	P@5	P@10	P@15	P@20	P@30
Indri	0.2851	0.2532	0.2170	0.2011	0.1801
MG4J	0.2480	0.2100	0.1800	0.1600	0.1340
Terrier	0.2800	0.2400	0.2130	0.2100	0.1930
Zettair	**0.3240**	**0.2680**	**0.2507**	**0.2310**	**0.1993**

- **Zettair: best average precision at top 5, 10, 20 results**

Global Evaluation

- Ranking of engines: indexing time, index size, query processing time (for 2.7GB collection), and P@5 (for WT10g collection)

Search Engine	Indexing Time (h:m:s)		Index Size (%)		Searching Time (ms)		Answer Quality P@5	
HtDig	(6)	0:28:30	(8)	104	(4)	32		-
Indri	(3)	0:15:45	(7)	63	(1)	**19**	(2)	0.2851
Lucene	(8)	1:01:25	(1)	**26**	(2)	21		-
MG4J	(2)	0:12:00	(6)	60	(3)	22	(4)	0.2480
Swish-E	(4)	0:19:45	(3)	31	(6)	45		-
Swish++	(5)	0:22:15	(2)	29	(8)	51		-
Terrier	(7)	0:40:12	(5)	52	(7)	50	(3)	0.2800
Zettair	(1)	**0:04:44**	(4)	33	(4)	32	(1)	**0.3240**

- Indri, MG4J, Terrier, and Zettair: indexed whole WT10g
- **Zettair: fastest indexer, good search time, good precision-recall**

Conclusions

- Zettair is one of the most complete engines
  1. fast processing of large amounts of information in considerably less time than other engines
  2. average precision-recall figures that were highest comparatively to the other engines (for the WT10g collection)
- Lucene is the most competitive regarding the use of memory and search time performance