NRC·CNRC

Canada Institute for Scientific and Technical Information

LuSql: (Quickly and easily)Getting your data from your DBMS into Lucene

Glen Newton

CISTI Research, CISTI NRC

code4lib 2009

Providence Rhode Island **Feb 24 2009**

Outline

- What is LuSql?
- Context
- Examples
- Performance & comparisons
- Next version

Context

- CISTI == Canada National Science Library
- Digital Library Research Group
- Heavy text mining, knowledge discovery tools, information visualization, citation analysis, recommender systems
- Large local text collection:
  - 8.4M PDFs, full text & metadata (~700GB)
  - Full text on file system
  - Metadata in MySql
- Team of 4: Lucene expert; 3 needing to use Lucene
- Daily creation of some experiment/domain/foo – specific large scale Lucene index

NRC·CNRC
Canada Institute for Scientific and Technical Information

LuSql Rationale

- Need for low barrier, high performance, flexible tool for Lucene index creation
- Choice
  - SOLR
  - DBSight
  - Lucen4DB.net
  - Hibernate Search
  - Compass
- All one or more of:
  - Overly complicated for non-Lucene / non-Java / non-XML / non-framework users
  - Performance/scalability issues
  - Not Open Source Software (OSS)

LuSql

- User knowledge:
  - Knowledge of SQL
  - Knowledge of their database and tables
  - Ability to set the Java CLASSPATH in a command-line shell
  - Ability to run a command line application

NRC·CNRC
Canada Institute for Scientific and Technical Information

LuSql Command Line Arguments

- Create or append
- SQL
- JDBC URL
- # records to index
- Lucene Analyzer class
- JDBC driver class
- Indexing properties, global or by field
- Global term value (i.e. all documents have "source=cat")
- Lucene RAM buffer size
- Stop word file
- Lucene index directory
- Multithreading toggle, #threads
- Pluggable Document filter
- Subqueries

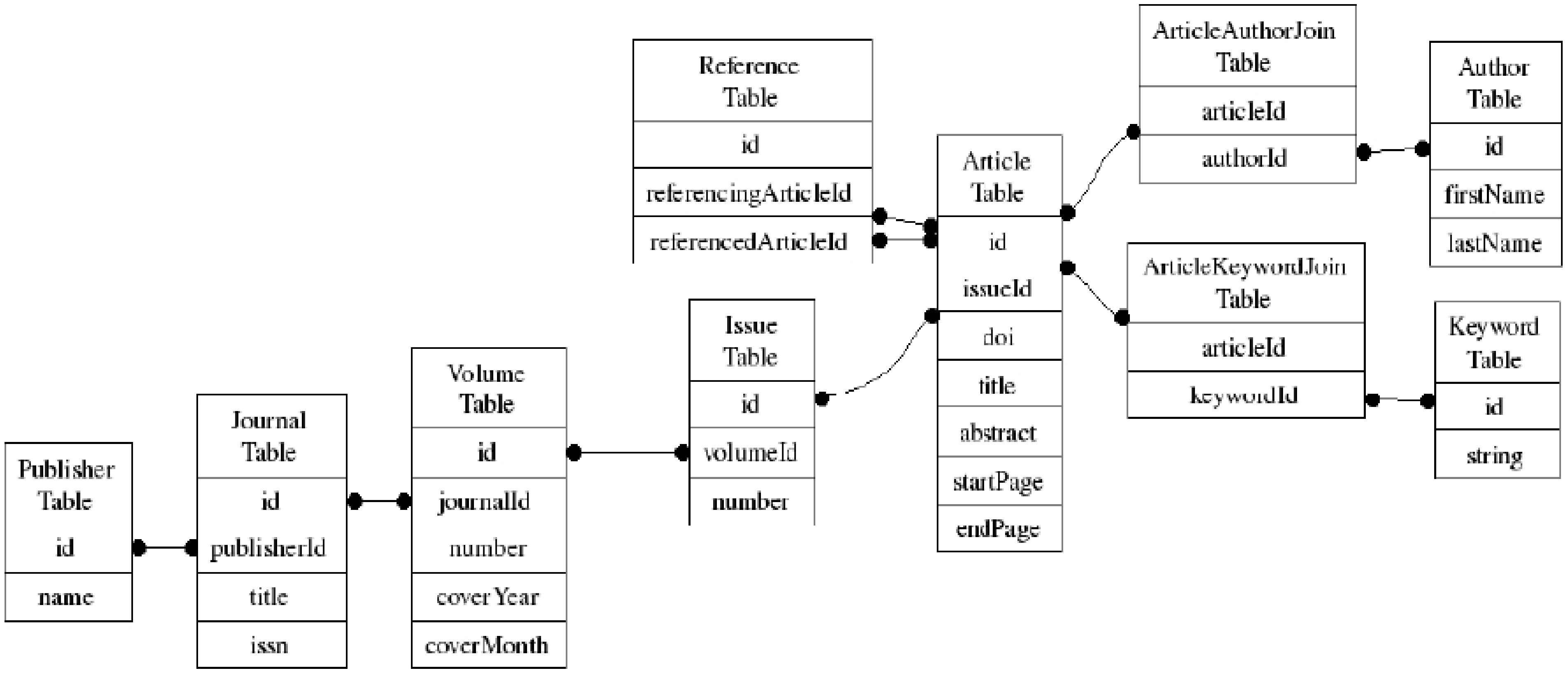


Figure 4: Table relationships in Journal Article database

Canada Institute for Scientific and Technical Information

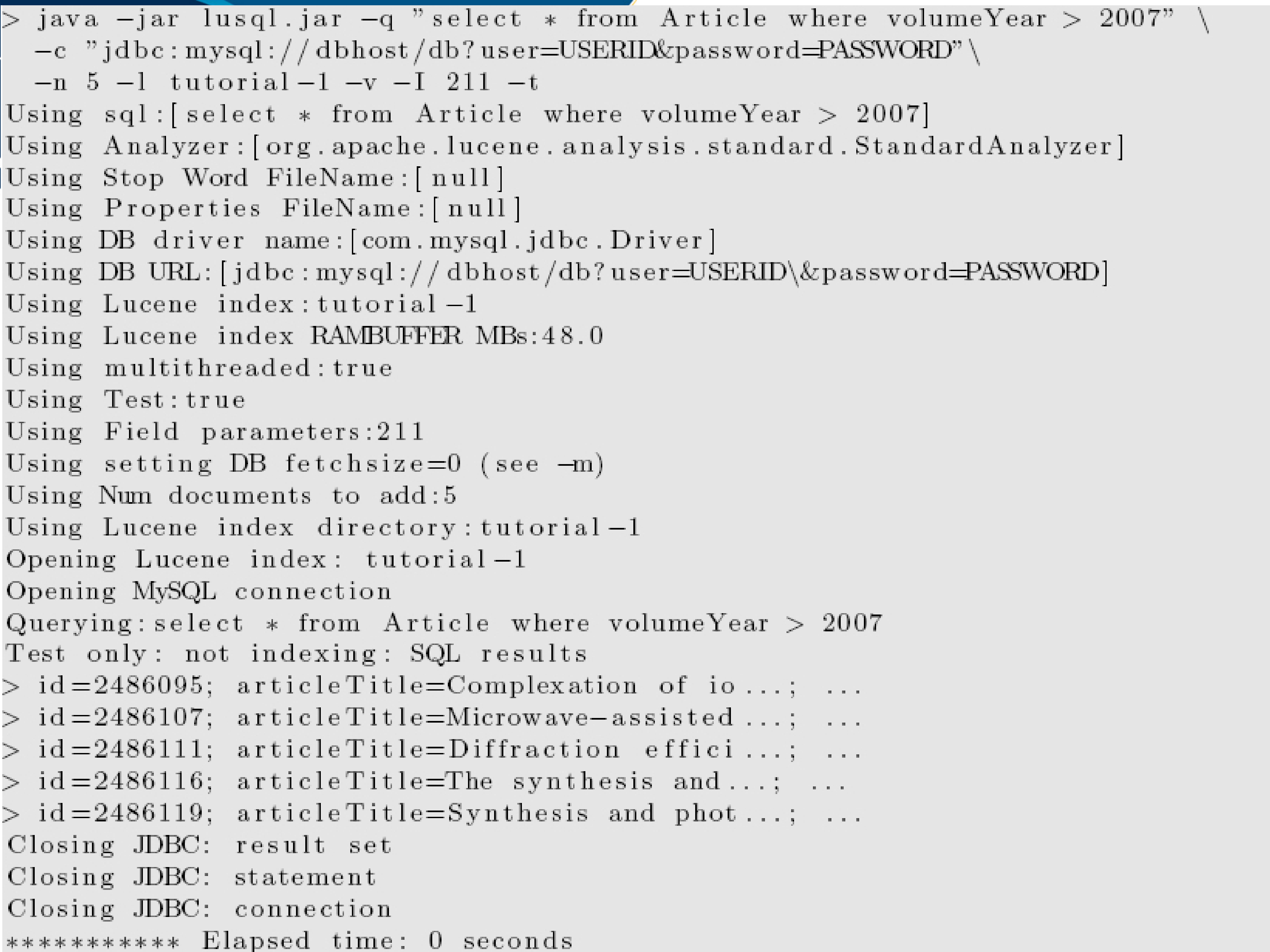
Examples

```
java jar lusql.jar -q "select * from Article where
  volumeYear > 2007" -c
  "jdbc:mysql://dbhost/db?user=ID&password=PASS" -n
  5 -l tutorial -I 211 -t
```

NRC·CNRC
Canada Institute for Scientific and Technical Information

Index Term Properties

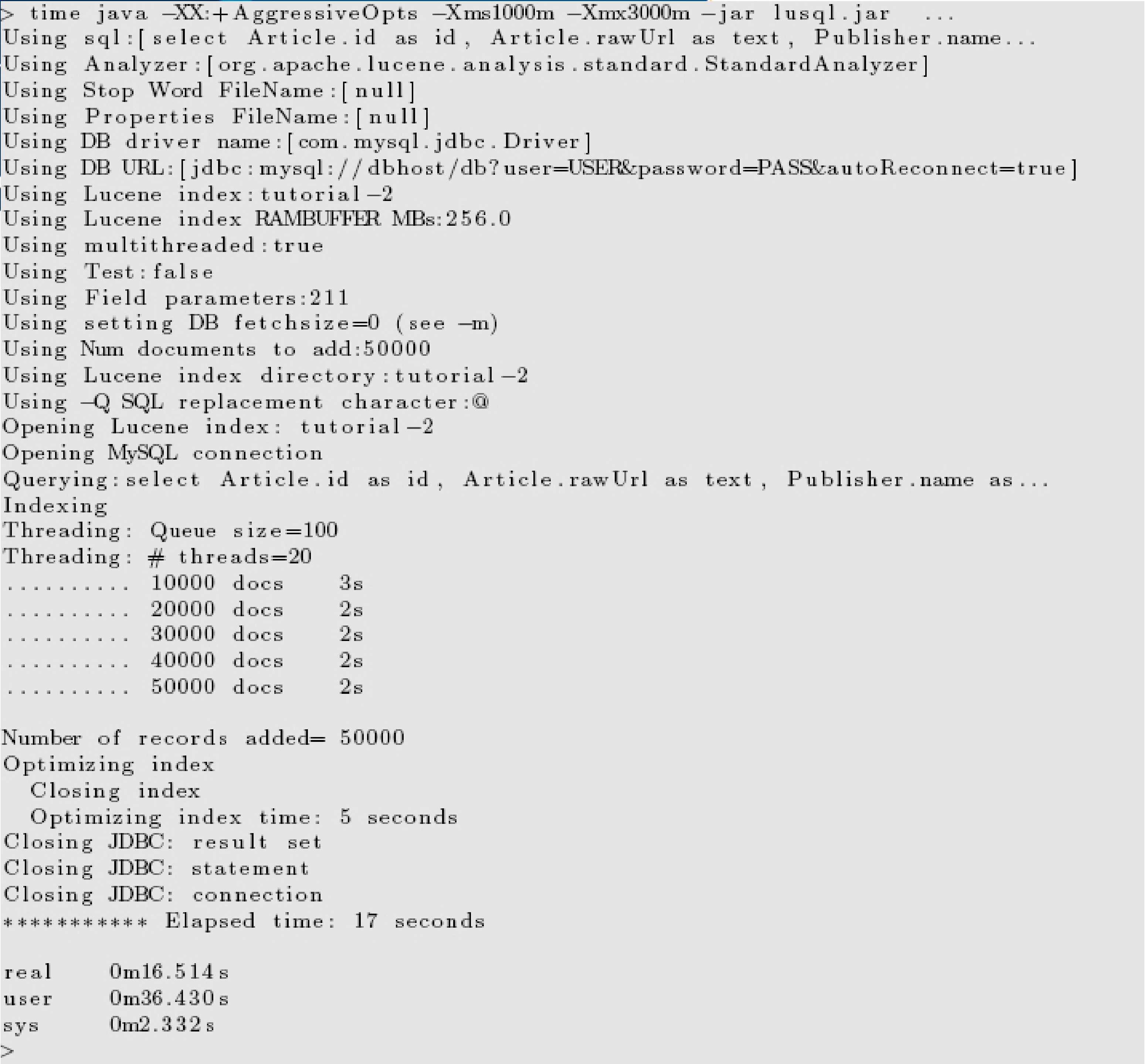
- Index: Default:TOKENIZED
  - 0:NO
  - 1:NO_NORMS
  - 2:TOKENIZED
  - 3:UN_TOKENIZED
- Store: Default:YES
  - 0:NO
  - 1:YES
  - 2:COMPRESS
- Term vector: Default:YES
  - 0:NO
  - 1:YES



```
> java -jar lusql.jar -q ”select * from Article where volumeYear > 2007” \
  -c ”jdbc:mysql://dbhost/db?user=USERID&password=PASSWORD” \
  -n 5 -l tutorial-1 -v -I 211 -t
Using sql:[select * from Article where volumeYear > 2007]
Using Analyzer:[org.apache.lucene.analysis.standard.StandardAnalyzer]
Using Stop Word FileName:[null]
Using Properties FileName:[null]
Using DB driver name:[com.mysql.jdbc.Driver]
Using DB URL:[jdbc:mysql://dbhost/db?user=USERID\&password=PASSWORD]
Using Lucene index:tutorial-1
Using Lucene index RAMBUFFER MBs:48.0
Using multithreaded:true
Using Test:true
Using Field parameters:211
Using setting DB fetchsize=0 (see -m)
Using Num documents to add:5
Using Lucene index directory:tutorial-1
Opening Lucene index: tutorial-1
Opening MySQL connection
Querying:select * from Article where volumeYear > 2007
Test only: not indexing: SQL results
> id=2486095; articleTitle=Complexation of io...; ...
> id=2486107; articleTitle=Microwave-assisted...; ...
> id=2486111; articleTitle=Diffraction effici...; ...
> id=2486116; articleTitle=The synthesis and...; ...
> id=2486119; articleTitle=Synthesis and phot...; ...
Closing JDBC: result set
Closing JDBC: statement
Closing JDBC: connection
*********** Elapsed time: 0 seconds
```

Example 2: Complex Join

```
java -jar lusql.jar -q "select Publisher.name as
  pub, Journal.title as jo,Article.rawUrl as text ,
  Journal.issn, Volume.number as
  vol,Volume.coverYear as year, Issue.number as
  iss, Article.id as id, Article.title as ti,
  Article.abstract as ab, Article.startPage as
  startPage, Article.endPage as endPage from
  Publisher, Journal, Volume, Issue, Article where
  Publisher.id = Journal.publisherId and Journal.id
  = Volume.journalId and Volume.id=Issue.volumeId
  and Issue.id = Article.issueId" -c "jdbc:mysql
  ://dbhost/db?user=ID&password=PASS" -n 50000 -l
  tutorial_2
```



```
> time java -XX:+AggressiveOpts -Xms1000m -Xmx3000m -jar lusql.jar  ...
Using sql:[select Article.id as id, Article.rawUrl as text, Publisher.name...
Using Analyzer:[org.apache.lucene.analysis.standard.StandardAnalyzer]
Using Stop Word FileName:[null]
Using Properties FileName:[null]
Using DB driver name:[com.mysql.jdbc.Driver]
Using DB URL:[jdbc:mysql://dbhost/db?user=USER&password=PASS&autoReconnect=true]
Using Lucene index:tutorial-2
Using Lucene index RAMBUFFER MBs:256.0
Using multithreaded:true
Using Test:false
Using Field parameters:211
Using setting DB fetchsize=0 (see -m)
Using Num documents to add:50000
Using Lucene index directory:tutorial-2
Using -Q SQL replacement character:@
Opening Lucene index: tutorial-2
Opening MySQL connection
Querying:select Article.id as id, Article.rawUrl as text, Publisher.name as...
Indexing
Threading: Queue size=100
Threading: # threads=20
.......... 10000 docs    3s
.......... 20000 docs    2s
.......... 30000 docs    2s
.......... 40000 docs    2s
.......... 50000 docs    2s

Number of records added= 50000
Optimizing index
  Closing index
  Optimizing index time: 5 seconds
Closing JDBC: result set
Closing JDBC: statement
Closing JDBC: connection
*********** Elapsed time: 17 seconds

real    0m16.514s
user    0m36.430s
sys     0m2.332s
>
```

Example 2: Large Scale Index Propertries

- 6.4M articles (only metadata)
- 20 fields, including abstract
- Indexing time: 1h 34m
- Index size: 21GB

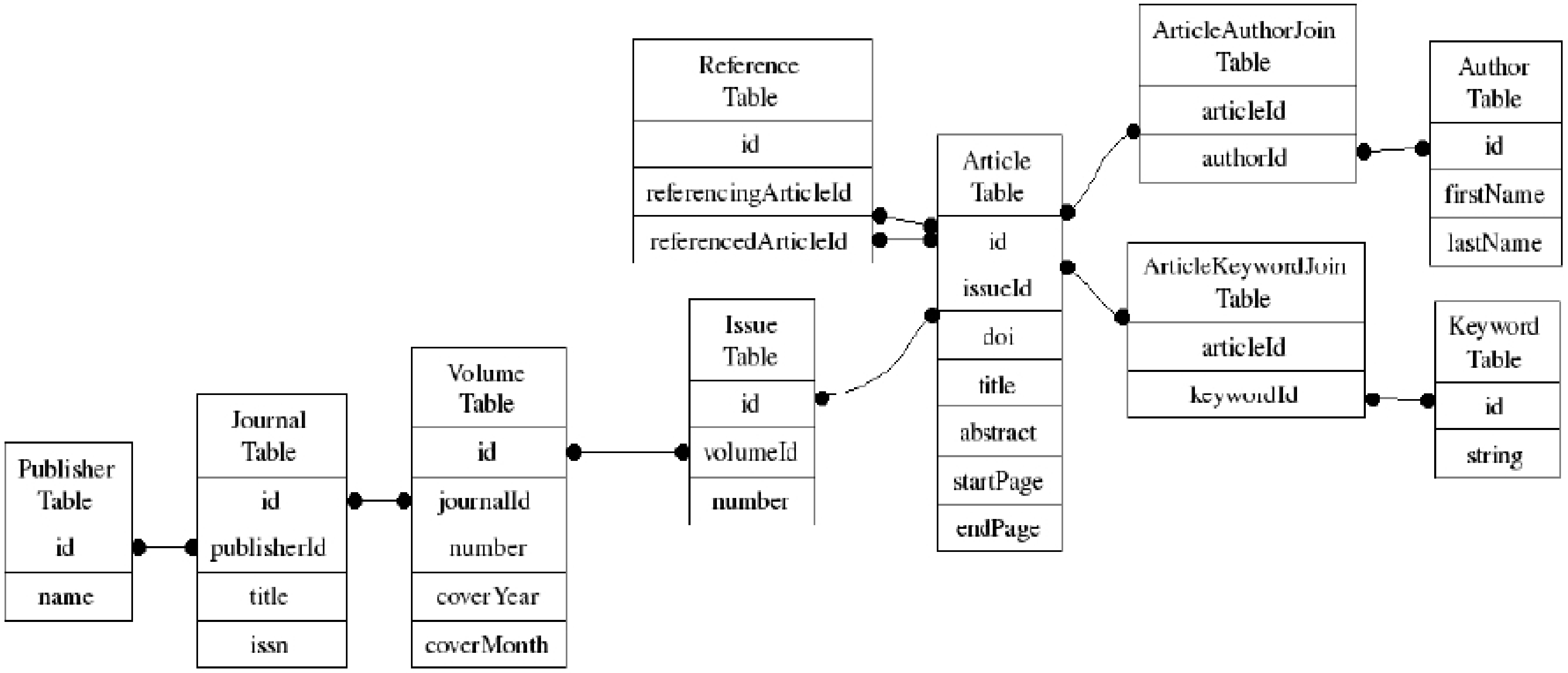


Figure 4: Table relationships in Journal Article database

The queries to select the appropriate authors, keyword and references for a particular article, say `Article.id=3453`, would therefor be:

```
select Keyword.string as keyword
 from ArticleKeywordJoin, Keyword
 where ArticleKeywordJoin.articleId=3453 and
 and ArticleKeywordJoin.keywordId = Keyword.id;

select concat(lastName,', ', firstName) as fullAuthor
 from ArticleAuthorJoin, Author
 where ArticleAuthorJoin.articleId = 3453
 and ArticleAuthorJoin.authorId = Author.id;

select referencedArticleId as citedId
 from Reference
 where Reference.referencingArticleId = 3453;
```

```
-Q "id|select Keyword.string as keyword from ArticleKeywordJoin, Keyword\
 where ArticleKeywordJoin.keywordId=@\
 and ArticleKeywordJoin.authorId = Keyword.id"\
-Q "id|select concat(lastName,', ', firstName) as fullAuthor\
 from ArticleAuthorJoin, Author where ArticleAuthorJoin.articleId = @\
```

Example 3: Complex Join

```
java -jar lusql.jar -q "select Publisher.name as
  pub, Journal.title as jo,Article.rawUrl as text ,
  Journal.issn, Volume.number as
  vol,Volume.coverYear as year, Issue.number as
  iss, Article.id as id, Article.title as ti,
  Article.abstract as ab, Article.startPage as
  startPage, Article.endPage as endPage from
  Publisher, Journal, Volume, Issue, Article where
  Publisher.id = Journal.publisherId and Journal.id
  = Volume.journalId and Volume.id=Issue.volumeId
  and Issue.id = Article.issueId" -c "jdbc:mysql
  ://dbhost/db?user=ID&password=PASS" -n 50000 -l
  tutorial_2
```

NRC·CNRC
Canada Institute for Scientific and Technical Information

Example 4: Out-of-band document manipulation

- Plugin architecture allowing arbitrary manipulations of `Documents` before they go into the index
- Implement `DocFilter` interface
- Add filter class at command line:
  - `-f ca.nrc.cisti.lusql.example.FileFullTextFilter`
  - Looks in metadata field for PDF location in file system; finds corresponding .txt file; reads file & adds to `Document`

NRC·CNRC
Canada Institute for Scientific and Technical Information

Example 4: Large Scale Index Properties

- 6.4M articles (metadata & full-text), ~600GB PDFs
- 21 fields, including abstract & full-text
- Indexing time: 13h 46m
- Index size: 86GB

Comparison to SOLR

- SOLR 1.4 November build:
  - Using DataImportHandler, with all defaults
- Lucene 2.4
- LuSql 0.90

NRC·CNRC
Canada Institute for Scientific and Technical Information

Hardware

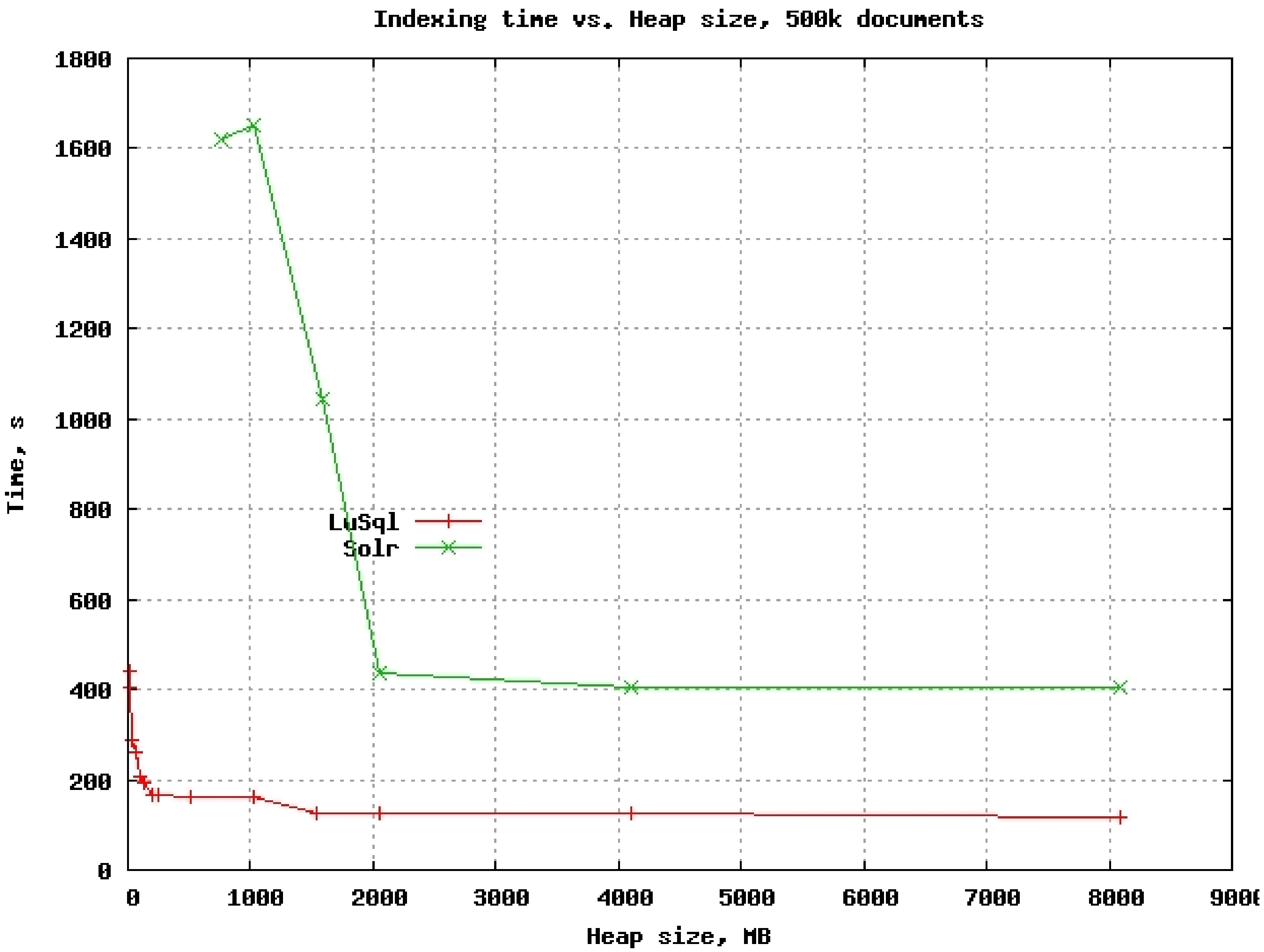
- Indexing and database machines:
  - Dell PowerEdge 1955 Blade server56
  - CPU: 2 x dual-core Xeon 5050 processors with 2x2MB cache, 3.0 Ghz 64bit
  - Memory: 8 GB 667MHz
  - Disk: 2 x 73GB internal 10K RPM SAS drives
- Both machines attached to:
  - Dell EMC AX150 storage arrays
  - 12 x 500 GB SATA II 7.2K RPM disks
  - via:
  - SilkWorm 200E57 Series 16-Port Capable 4Gb Fabric Switch

NRC·CNRC
Canada Institute for Scientific and Technical Information

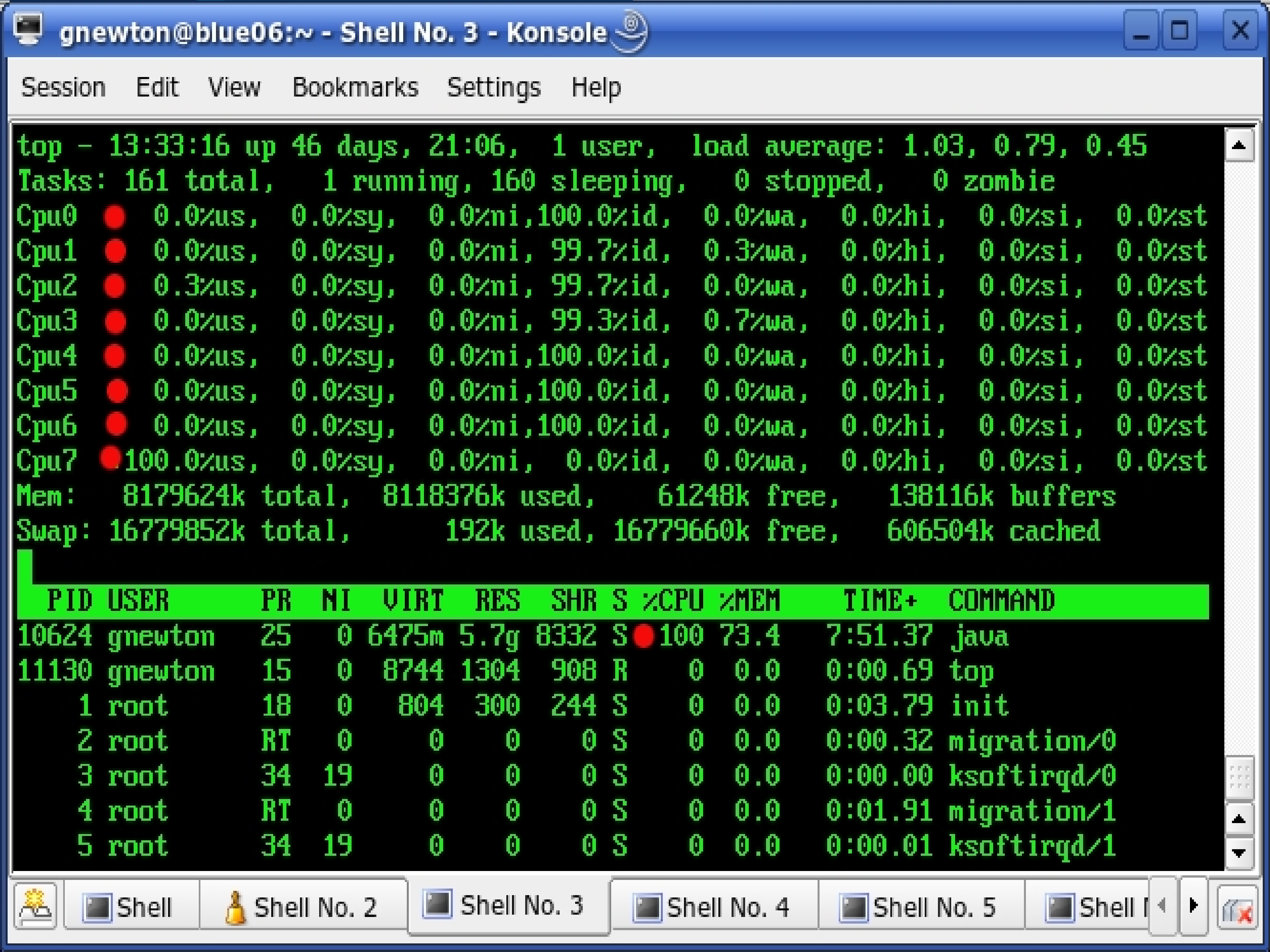
Software

- MySql: v5.0.45 compiled from source.
- gcc: gcc version 4.1.2 20061115 (prerelease) (SUSE Linux)
- Java: java version 1.6.0 07 SE Runtime Environment (build 1.6.0 07-b06) Java HotSpot(TM) 64-Bit Server VM (build 10.0-b23, mixed mode)
- Operating System
  – Linux openSUSE 10.2 (64-bit X86-64)
  – Linux kernel: 2.6.18.8-0.10-default #1 SMP

	LuSql		SOLR	
Heap	RamBuffer, MB	Indexing Time, s	RamBuffer MB	Indexing Time, s
18	2	440	-	-
24	3	403	-	-
32	8	288	-	-
64	16	263	-	-
96	24	209	-	-
128	32	193	-	-
192	64	168	-	-
256	96	167	-	-
512	128	161	-	-
768	384	165	4	1621
1024	384	162	32	1651
1536	384	126	128	1045
2048	512	126	512	438
4096	3072	124	1024	404
8096	4096	119	1024	407



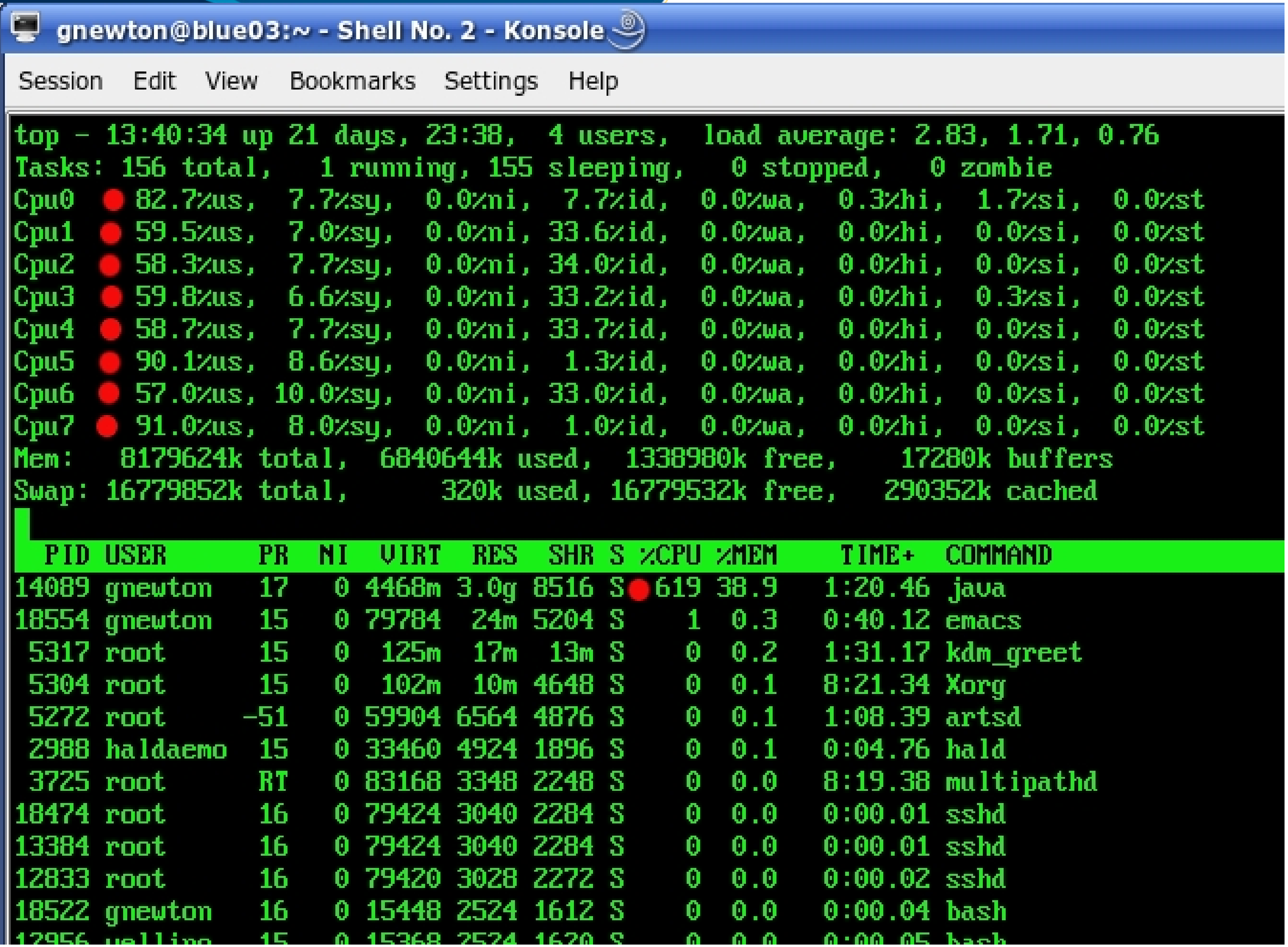
Indexing time vs. Heap size, 500k documents
Time, s
Heap size, MB
LuSql
Solr
0
200
400
600
800
1000
1200
1400
1600
1800
1000
2000
3000
4000
5000
6000
7000
8000
900



gnewton@blue06:~ - Shell No. 3 - Konsole
Session Edit View Bookmarks Settings Help
Shell
Shell No. 2
Shell No. 3
Shell No. 4
Shell No. 5
Shell

```
top - 13:33:16 up 46 days, 21:06,  1 user,  load average: 1.03, 0.79, 0.45
Tasks: 161 total,   1 running, 160 sleeping,   0 stopped,   0 zombie
Cpu0 ●  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1 ●  0.0%us,  0.0%sy,  0.0%ni, 99.7%id,  0.3%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2 ●  0.3%us,  0.0%sy,  0.0%ni, 99.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3 ●  0.0%us,  0.0%sy,  0.0%ni, 99.3%id,  0.7%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu4 ●  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu5 ●  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu6 ●  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu7 ●100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   8179624k total,  8118376k used,    61248k free,   138116k buffers
Swap: 16779852k total,      192k used, 16779660k free,   606504k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
10624 gnewton   25   0 6475m 5.7g 8332 S ●100 73.4   7:51.37 java
11130 gnewton   15   0  8744 1304  908 R    0  0.0   0:00.69 top
    1 root      18   0   804  300  244 S    0  0.0   0:03.79 init
    2 root      RT   0     0    0    0 S    0  0.0   0:00.32 migration/0
    3 root      34  19     0    0    0 S    0  0.0   0:00.00 ksoftirqd/0
    4 root      RT   0     0    0    0 S    0  0.0   0:01.91 migration/1
    5 root      34  19     0    0    0 S    0  0.0   0:00.01 ksoftirqd/1
```



```
top - 13:40:34 up 21 days, 23:38,  4 users,  load average: 2.83, 1.71, 0.76
Tasks: 156 total,   1 running, 155 sleeping,   0 stopped,   0 zombie
Cpu0  ● 82.7%us,  7.7%sy,  0.0%ni,  7.7%id,  0.0%wa,  0.3%hi,  1.7%si,  0.0%st
Cpu1  ● 59.5%us,  7.0%sy,  0.0%ni, 33.6%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  ● 58.3%us,  7.7%sy,  0.0%ni, 34.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  ● 59.8%us,  6.6%sy,  0.0%ni, 33.2%id,  0.0%wa,  0.0%hi,  0.3%si,  0.0%st
Cpu4  ● 58.7%us,  7.7%sy,  0.0%ni, 33.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu5  ● 90.1%us,  8.6%sy,  0.0%ni,  1.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu6  ● 57.0%us, 10.0%sy,  0.0%ni, 33.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu7  ● 91.0%us,  8.0%sy,  0.0%ni,  1.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   8179624k total,  6840644k used,  1338980k free,    17280k buffers
Swap: 16779852k total,      320k used, 16779532k free,   290352k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
14089 gnewton   17   0 4468m 3.0g 8516 S ●619 38.9   1:20.46 java
18554 gnewton   15   0 79784  24m 5204 S    1  0.3   0:40.12 emacs
 5317 root      15   0  125m  17m  13m S    0  0.2   1:31.17 kdm_greet
 5304 root      15   0  102m  10m 4648 S    0  0.1   8:21.34 Xorg
 5272 root     -51   0 59904 6564 4876 S    0  0.1   1:08.39 artsd
 2988 haldaemo  15   0 33460 4924 1896 S    0  0.1   0:04.76 hald
 3725 root      RT   0 83168 3348 2248 S    0  0.0   8:19.38 multipathd
18474 root      16   0 79424 3040 2284 S    0  0.0   0:00.01 sshd
13384 root      16   0 79424 3040 2284 S    0  0.0   0:00.01 sshd
12833 root      16   0 79420 3028 2272 S    0  0.0   0:00.02 sshd
18522 gnewton   16   0 15448 2524 1612 S    0  0.0   0:00.04 bash
12956 yelling   15   0 15368 2524 1620 S    0  0.0   0:00.05 bash
```

NRC·CNRC
Canada Institute for Scientific and Technical Information

Acknowledgments

- Greg Kresko, Andre Vellino, Jeff Demaine, various LuSql users

NRC·CNRC
Canada Institute for Scientific and Technical Information

LuSql 0.95 in development

- Re-architected to have pluggable drivers for both input & output
- Read drivers:
  - **JDBC**, **Lucene**, Minion, **BDB**. ehcache, SolrJ, RMI, Terrier
- Write drivers:
  - **JDBC**, **Lucene**, Minion, **BDB**. ehcache, SolrJ, **text**, **XML**, RMI, Terrier
- For large volumes, concurrent multiple indexes merged at end

NRC·CNRC
Canada Institute for Scientific and Technical Information

Questions

- Glen Newton glen.newton@nrc-cnrc.gc.ca

NRC·CNRC

Canada Institute for Scientific and Technical Information

Science at work for Canada