Effective Keyword Search for Software Resources installed in Large-scale Grid Infrastructures

George Pallis, Asterios Katsifodimos, Marios D. Dikaiakos
University of Cyprus
Computing Grids

- Distributed computing infrastructures that enable flexible, secure, coordinated resource sharing among dynamic collections of individuals and institutions (Foster, Kesselman, Tuecke).
- Enable communities ("Virtual Organizations") to share geographically distributed resources as they pursue common goals --
- Key assumptions: absence of...
  - Homogeneity
  - Central location
  - Central control
  - Existing trust relationships
Computational Grids

- Nowadays, Grid infrastructures comprise an impressive collection of computational and software resources.
Computational Grids

- Nowadays, Grid infrastructures comprise an impressive collection of computational and software resources.
How can we search for software that is installed on the sites of a large-scale Grid infrastructure?
Software resources and services need to be easily discoverable by and accessible to end-users.
Software resources and services need to be easily discoverable by and accessible to end-users to enhance
Software resources and services need to be easily discoverable by and accessible to end-users to enhance inquiries about infrastructure functionality.
Software resources and services need to be easily discoverable by and accessible to end-users to enhance inquiries about infrastructure functionality software reuse
Software resources and services need to be easily discoverable by and accessible to end-users to enhance inquiries about infrastructure functionality.

Software reuse

Resource selection
Software resources and services need to be easily discoverable by and accessible to end-users to enhance inquiries about infrastructure functionality, software reuse, resource selection, low-cost entry to the infrastructure.
What are the options?
Go manual
Go manual

In EGEE, a user would have to gain access and search inside the file systems of 267 sites, several of which host well over 1 million software-related files.
Go manual

- In EGEE, a user would have to gain access and search inside the file systems of 267 sites, several of which host well over 1 million software-related files.
- Direct access is impossible, for security reasons.
Go manual

- In EGEE, a user would have to gain access and search inside the file systems of **267 sites**, several of which host well **over 1 million** software-related files.
- Direct access is impossible, for security reasons.
- “grep” does not provide good answers, especially if one is looking for generic information (“**find graph analysis software**”).
Go manual

- In EGEE, a user would have to gain access and search inside the file systems of **267 sites**, several of which host well **over 1 million** software-related files.
- Direct access is impossible, for security reasons.
- “grep” does not provide good answers, especially if one is looking for generic information (**“find graph analysis software”**).
- Traditional file systems provide limited metadata about file types and relationships.
Go manual

- In EGEE, a user would have to gain access and search inside the file systems of 267 sites, several of which host well over 1 million software-related files
- Direct access is impossible, for security reasons
- “grep” does not provide good answers, especially if one is looking for generic information ("find graph analysis software")
- Traditional file systems provide limited metadata about file types and relationships
- Semantic file systems do exist but are not widely adopted
Go Google
Go Google

Software is not transcribed in HTML, XML, or anything close to natural language
Go Google

- Software is not transcribed in HTML, XML, or anything close to natural language
- Files are not accessible via HTTP
Go Google

- Software is not transcribed in HTML, XML, or anything close to natural language
- Files are not accessible via HTTP
- No embedded hyperlinks that could help with result ranking
Search inside a database (white pages)
Search inside a database (white pages)

- Grid Information Services provide some query facilities (LDAP, SQL) but store little, if any, tags about installed software
Search inside a database (white pages)

- Grid Information Services provide some query facilities (LDAP, SQL) but store little, if any, tags about installed software
- Tag setup is *manual* and often not done at all
Search inside a database (white pages)

- Grid Information Services provide some query facilities (LDAP, SQL) but store little, if any, tags about installed software
- Tag setup is manual and often not done at all
- Modeling Grid-related information is not trivial
Search inside a database (**white pages**)

- Grid Information Services provide some query facilities (LDAP, SQL) but store little, if any, tags about installed software
- Tag setup is *manual* and often not done at all
- Modeling Grid-related information is not trivial

Some intrinsic challenges
Some intrinsic challenges

- Software installation and management in federated distributed computing infrastructures is performed:
Some intrinsic challenges

- Software installation and management in federated distributed computing infrastructures is performed:
  - in a decentralized, often uncoordinated, manner
  - by various actors (sysadmins, VO managers, application developers, end-users)
  - without following some common standard for software packaging and description
Some intrinsic challenges

- Software installation and management in federated distributed computing infrastructures is performed:
  - in a decentralized, often uncoordinated, manner
  - by various actors (sysadmins, VO managers, application developers, end-users)
  - without following some common standard for software packaging and description

- Software installed in Grid infrastructures is...
Some intrinsic challenges

- Software installation and management in federated distributed computing infrastructures is performed:
  - in a decentralized, often uncoordinated, manner
  - by various actors (sysadmins, VO managers, application developers, end-users)
  - without following some common standard for software packaging and description

- Software installed in Grid infrastructures is:
  - found in various formats (sources, binaries, libraries)
  - located in unstructured repositories (file systems), alongside numerous resources of various types
Goal

- Provide an engine that would enable full-text search for software installed on Grids and Clouds: Minersoft
Goal

- Provide an engine that would enable full-text search for software installed on Grids and Clouds: **Minersoft**

Minersoft

Octave Numerical Computations

- search

1. time.oct [Grid sites]
2. rand.oct [Grid sites]
3. sqrtm.oct [Grid sites]
4. dasrt_options.oct [Grid sites]
5. emacs [Grid sites]
6. sox [Grid sites]
7. ...
Our approach and contribution
Our approach and contribution

- **Harvest** the file system of Grid computing sites for installed software-related files and their inter-relationships
Our approach and contribution

- **Harvest** the file system of Grid computing sites for installed software-related files and their inter-relationships
- Introduce the **Software Graph**, a data structure used to represent software-related files and their inter-relationships
Our approach and contribution

- **Harvest** the file system of Grid computing sites for installed software-related files and their inter-relationships
- Introduce the **Software Graph**, a data structure used to represent software-related files and their inter-relationships
- Use the Software Graph to “enrich” software-related files with descriptive keywords extracted from the files’ context
Our approach and contribution

- **Harvest** the file system of Grid computing sites for installed software-related files and their inter-relationships
- Introduce the **Software Graph**, a data structure used to represent software-related files and their inter-relationships
- Use the Software Graph to “enrich” software-related files with descriptive keywords extracted from the files’ context
- Build inverted indexes to support text-based software retrieval
Our approach and contribution

- **Harvest** the file system of Grid computing sites for installed software-related files and their inter-relationships
- Introduce the **Software Graph**, a data structure used to represent software-related files and their inter-relationships
- Use the Software Graph to “enrich” software-related files with descriptive keywords extracted from the files’ context
- Build inverted indexes to support text-based software retrieval
- Evaluate the quality of Minersoft’s results using a real Grid testbed
## Related Work on Software Retrieval

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Corpus</th>
<th>Search paradigm</th>
<th>Software resources</th>
<th>Binaries</th>
<th>Source Codes</th>
<th>Description Docs</th>
<th>Binary Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>GURU</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SEC</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACM SAC, 2006</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maracatu</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACM SAC, 2007</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extreme Harvesting</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IEEE IRI, 2004</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPARS-J</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Koders</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Google Code Search</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sourcerer</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMKD, 2009</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minersoft</td>
<td>Grid/Cloud</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Related Work on Software Retrieval

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Corpus</th>
<th>Search paradigm</th>
<th>Binaries</th>
<th>Source Codes</th>
<th>Description Docs</th>
<th>Binary Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>GURU</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SEC</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACM SAC, 2006</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maracatu</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACM SAC, 2007</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extreme Harvesting</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IEEE IRI, 2004</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPARS-J</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Koders</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Google Code Search</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sourcecerer</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMKD, 2009</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minersoft</td>
<td>Grid/Cloud</td>
<td>Keyword-based</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Related Work on Software Retrieval

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Corpus</th>
<th>Search paradigm</th>
<th>Software resources</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Binaries</td>
</tr>
<tr>
<td>GURU</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
<tr>
<td>SEC</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
<tr>
<td>ACM SAC, 2006</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maracatu</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
<tr>
<td>ACM SAC, 2007</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extreme Harvesting</td>
<td>Web</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
<tr>
<td>IEEE IRI, 2004</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPARS-J</td>
<td>Web</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
<tr>
<td>Koders</td>
<td>Web</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
<tr>
<td>Google Code Search</td>
<td>Web</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
<tr>
<td>Source器er</td>
<td>Web</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
<tr>
<td>DMKD, 2009</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minersoft</td>
<td>Grid/Cloud</td>
<td>Keyword-based</td>
<td>•</td>
</tr>
</tbody>
</table>

## Related Work on Software Retrieval

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Corpus</th>
<th>Search paradigm</th>
<th>Software resources</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Binaries</td>
</tr>
<tr>
<td><strong>GURU</strong></td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
<tr>
<td><strong>SEC</strong></td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
<tr>
<td><em>ACM SAC, 2006</em></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Maracatu</strong></td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
<tr>
<td><em>ACM SAC, 2007</em></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Extreme Harvesting</strong></td>
<td>Web</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
<tr>
<td><em>IEEE IRI, 2004</em></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SPARS-J</strong></td>
<td>Web</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
<tr>
<td><strong>Koders</strong></td>
<td>Web</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
<tr>
<td><strong>Google Code Search</strong></td>
<td>Web</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
<tr>
<td><strong>Sourcerer</strong></td>
<td>Web</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
<tr>
<td><em>DMKD, 2009</em></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Minersoft</strong></td>
<td>Grid/Cloud</td>
<td>Keyword-based</td>
<td>●</td>
</tr>
</tbody>
</table>

### Related Work on Software Retrieval

<table>
<thead>
<tr>
<th>Approaches</th>
<th>Corpus</th>
<th>Search paradigm</th>
<th>Software resources</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Binaries</td>
</tr>
<tr>
<td>GURU</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
</tr>
<tr>
<td>SEC</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
</tr>
<tr>
<td>ACM SAC, 2006</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maracatu</td>
<td>Software Repositories</td>
<td>Keyword-based</td>
<td></td>
</tr>
<tr>
<td>ACM SAC, 2007</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extreme Harvesting</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
</tr>
<tr>
<td>IEEE IRI, 2004</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPARS-J</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
</tr>
<tr>
<td>Koders</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
</tr>
<tr>
<td>Google Code Search</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
</tr>
<tr>
<td>Sourceer</td>
<td>Web</td>
<td>Keyword-based</td>
<td></td>
</tr>
<tr>
<td>DMKD, 2009</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minersoft</td>
<td>Grid/Cloud</td>
<td>Keyword-based</td>
<td></td>
</tr>
</tbody>
</table>
Outline

- Motivation, context, related work
- **Minersoft: data structure and algorithm**
- Architecture and implementation
- Evaluation
- Conclusions
Software Resource

A software resource is a file that belongs to one of the following categories:

- Executables (binaries or scripts)
- Software libraries
- Source codes
- Unstructured or semi-structured software-description and software-configuration documents (manuals, readme files, makefiles RPMs)
Software Graph

- Software Graph is a **weighted, metadata-rich, typed** graph $G(V,E)$
Software Graph

- Each vertex $v$ is annotated with associated metadata attributes that describe $v$'s content and context.
Software Graph

- Each vertex $v$ is annotated with associated metadata attributes that describe $v$’s content and context.
- Each edge $e$ has a type and a weight.

```
name
  type
  site
  path
  zones

/  bin  lib  tar  libtar.so  libgzip.so  tar-2.6  tar-2.4.3  Readme  Readme  Readme  gzip  gzip.h
```

type $(e)$

$w(e)$ (0 < $w$ ≤ 1)
Minersoft Processing
Minersoft Processing

**File System Harvesting**
Minersoft Processing

File System Harvesting → File Type Classification
Minersoft Processing

File System Harvesting → File Type Classification → Software Graph Construction
Minersoft Processing

- File System Harvesting
- File Type Classification
- Software Graph Construction
- Full-text Indexing
Minersoft Processing

1. File System Harvesting
2. File Type Classification
3. Software Graph Construction
4. Full-text Indexing
5. Query Processing
Minersoft Processing

File System Harvesting → File Type Classification → Software Graph Construction → Full-text Indexing → Query Processing
FST construction

- Scan the file-system and map the FST
  - Edges denote containment relationships inside the file-system
  - Stop lists
Classification and pruning

- Normalize file-names
- Extract path-names
- Apply expert rules and use syscalls to classify files
- Drop irrelevant files and childless directories
Structural-dependency elicitation
Structural-dependency elicitation

- Goal: to enrich the “context” of each software-related file
Structural-dependency elicitation

- Goal: to enrich the “context” of each software-related file
- Discover FST-vertex relationships that emanate from “structural” properties of the FST vertices, besides directory containment:
  - naming and location conventions
  - file-system organization practices about software documentation
Structural-dependency elicitation

- Goal: to enrich the “context” of each software-related file
- Discover FST-vertex relationships that emanate from “structural” properties of the FST vertices, besides directory containment:
  - naming and location conventions
  - file-system organization practices about software documentation
- Looking for:
  - Documentation files describing software via 1-1 (man pages, javadocs) or 1-many relationships (readme’s)
  - Dependencies between libraries and binary executables
Structural-dependency elicitation
Keyword scrapping

- Parse the content of SG vertices and extract descriptive keywords
  - Assign them to the vertex “content zone”
- Content parsing
- Stop-word elimination
- Stemming
- Keyword extraction
- Computationally demanding and data-intensive task
Keyword flow
Keyword flow

- Goal: to enrich the content of software-file vertices of the SG with relevant keywords
Keyword flow

- Goal: to enrich the content of software-file vertices of the SG with *relevant* keywords
- If there is an edge in SG between a software-documentation vertex and a software-file vertex
Keyword flow

- Goal: to enrich the content of software-file vertices of the SG with *relevant* keywords
- If there is an edge in SG between a software-documentation vertex and a software-file vertex
  - Copy keywords from the documentation-vertex to its associated software-file vertices
Keyword flow

- Goal: to enrich the content of software-file vertices of the SG with relevant keywords
- If there is an edge in SG between a software-documentation vertex and a software-file vertex
  - Copy keywords from the documentation-vertex to its associated software-file vertices
  - Enrich the content of the software file by adding a new zone for each doc2softw edge of the SG
Keyword flow

- Goal: to enrich the content of software-file vertices of the SG with *relevant* keywords

- If there is an edge in SG between a software-documentation vertex and a software-file vertex
  - Copy keywords from the documentation-vertex to its associated software-file vertices
  - Enrich the content of the software file by adding a new zone for each doc2softw edge of the SG
  - Zone gets the weight of the SG’s edge, normalized so that the sum of the zone-weights of each SG vertex is equal to one.
Inverted index construction & merge

<table>
<thead>
<tr>
<th>terms</th>
<th>postings</th>
</tr>
</thead>
<tbody>
<tr>
<td>winzip</td>
<td>1,2,....</td>
</tr>
<tr>
<td>octave</td>
<td>3,6,....</td>
</tr>
<tr>
<td>....</td>
<td>.....</td>
</tr>
</tbody>
</table>
Outline

- Motivation, context, related work
- Minersoft structures and algorithm
- Architecture and implementation
- Evaluation
- Conclusions
Minersoft  Harvester Challenges

- Take advantage of Grid computing and storage
  - Distribute parts of computation to Grid sites
- Avoid overloading Grid sites
  - Apply load-balanced techniques
- Overlap local computation with I/O
  - Employ multi-threading jobs
- Adapt to the Grid sites policies
  - Number of jobs in queuing systems
  - Total time that a job is allowed to run on a given site
Minersoft Architecture

- Map-reduce like architecture
- Harvesting and indexing is done by multithreaded Grid jobs
- The harvester and indexer jobs process a specific number of files => splits
- Split size:
  - Each split is chosen so that the crawling and indexing can be distributed within the system time constraints
- Duplicate elimination

Minersoft Architecture

- Supervision of crawler and indexer Grid jobs
- Fetches the results from storage elements of Grid sites
The results of crawlers and indexers are stored at the storage elements of each Grid site.
Minersoft Architecture

Constructs the Software Graph

- Users
- Query Processor
- Global file index
- Global inverted index
- Graph Constructor
- Grid Job Manager
- Batch system queue
- Grid
  - Grid Site 1
  - Max Wall Clock Time
  - File index
  - Full text inverted index
  - Batch system queue
  - Grid Site 2
  - Max Wall Clock Time
  - File index
  - Full text inverted index
  - Batch system queue
  - Grid Site N
  - Max Wall Clock Time
  - File index
  - Full text inverted index
Minersoft Architecture

Receives the results and ranks them

Users

Minersoft

Query Processor

Global file index

Global inverted index

Graph Constructor

Grid Job Manager

Batch system queue

Grid

Batch system queue

Grid Site 1
Max Wall Clock Time

File index

Full text inverted index

Batch system queue

Grid Site 2
Max Wall Clock Time

File index

Full text inverted index

Batch system queue

Grid Site N
Max Wall Clock Time

File index

Full text inverted index
### Implementation details

<table>
<thead>
<tr>
<th>Component</th>
<th>Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grid job manager</td>
<td>Ganglia</td>
</tr>
<tr>
<td>Crawler</td>
<td>Python</td>
</tr>
<tr>
<td>Indexer</td>
<td>Lucene</td>
</tr>
</tbody>
</table>
Outline

- Motivation, context, related work
- Minersoft structures and algorithm
- Architecture and implementation
- Evaluation
- Conclusions
## Testbed

<table>
<thead>
<tr>
<th>Grid Site</th>
<th>Number of Software Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>AEGIS01-PHY-SCL</td>
<td>120,369</td>
</tr>
<tr>
<td>CY-03-INTERCOLLEGE</td>
<td>72,424</td>
</tr>
<tr>
<td>CY-01-KIMON</td>
<td>565,799</td>
</tr>
<tr>
<td>RO-08-UVT</td>
<td>157,591</td>
</tr>
<tr>
<td>HG-05-FORTH</td>
<td>1,508,986</td>
</tr>
<tr>
<td>BG04-ACAD</td>
<td>2,632,193</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>5,057,362</strong></td>
</tr>
</tbody>
</table>
Evaluation

- Data collection
  - 6 Grid sites of EGEE infrastructure

- Queries
  - 27 queries, (provided by EGEE users)

- Relevance Judgment
  - Satisfied
  - Not satisfied
  - Very satisfied
Evaluation

- Data collection
  - 6 Grid sites of EGEE infrastructure

- Queries
  - 27 queries, (provided by EGEE users)

- Relevance Judgment
  - Satisfied
  - Not satisfied
  - Very satisfied

<table>
<thead>
<tr>
<th>General-content queries</th>
<th>Software-specific queries</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear algebra package; fast fourier transformations; symbolic algebra computation library; mathematics statistics analysis; earthquake analysis; scientific data processing; statistical analysis software; atlas software</td>
<td>ImageMagick; lapack library; GSL library; crab; k3b cd burning; xerces xml; gcc fortran; octave numerical computations; matlab; hpc netlib; scalapack; mpich; autodock docking; boost c++ library; subversion client; java virtual machine; ffmpeg video processing; FFTW library</td>
</tr>
</tbody>
</table>
Benchmark Data Set

Welcome to Minersoft’s software retrieval evaluation dataset page

Here, you can find the dataset used to evaluate Minersoft’s performance in Software Retrieval.

Indexes

The files provided are inverted indexes built with Apache Lucene.
Navigate to the indexes directory for a list of available files:
- Indexes

We provide 18 inverted indexes. There are another 2, which we do not provide because they are very large to be provided through a web server. However (for those who are interested to get them) we can provide them upon request. The indexes contain many different zones (a.k.a Lucene fields). You can use the Lucene package to get familiar with the different fields. You will need some Lucene code to be able to query the indexes.

Relevance Judgements

We provide the relevance judgements that we used to evaluate Minersoft’s software retrieval performance. We use 28 queries and a relevance value ranging from 0-2 (0 not relevant, 1 relevant, 2 more relevant). We used the NDCG and HCG information retrieval metrics. In the zip file you will find two excel spreadsheets. One for the queries using the stemmed fields and one for the non-stemmed fields:
- Judgments zip

Enquiries

Please contact Asterios Katsifodimos (asteriosk@cs.ucy.ac.cy) for any further information/details or feedback.

https://thales.grid.ucy.ac.cy/minersoft/dataset/
Examined Metrics

<table>
<thead>
<tr>
<th>name</th>
<th>type</th>
<th>site</th>
<th>path</th>
</tr>
</thead>
</table>

- Content zone
- Doc. zones
- Norm. text zones
Examined Metrics

- Precision@20
- Normalized Discounted Cumulative Gain (NDCG)
Examined Metrics

- Precision@20
- Normalized Discounted Cumulative Gain (NDCG)
- Normalized Cumulative Gain (NCG)
Examined Metrics

- Precision@20
- Normalized Discounted Cumulative Gain (NDCG)
- Normalized Cumulative Gain (NCG)

Scenarios:
- File-search
  - full-text content of discovered files
- Context-enhanced search
- Software-description-enriched search
- Text-file-enriched search
Experimental Evaluation

![Graph showing precision at 20 (Precision@20) for different search types and query categories. The graph compares Total Queries, General-content Queries, and Software-specific Queries. The categories include File-search, Context-enhanced search, Software-description enriched-search, and Text-file-enriched search.]
Experimental Evaluation

![Bar Chart]

- **NDCG**
  - Total Queries
  - General-content Queries
  - Software-specific Queries

- **Methods**
  - Average
  - Median

- **Categories**
  - File-search
  - Context-enhanced-search
  - Software-description-enriched-search
  - Text-file-enriched-search
Experimental Evaluation
Conclusion – Future Work

- Minersoft - a Grid harvester which enables keyword-based searches for software installed on Grid computing infrastructures
- The results of Minersoft harvesting are encoded in a weighted typed graph, called the Software Graph
- **Ongoing work**: to extend Minersoft so as to support search on Cloud infrastructures
- **Future work**: to represent software resources using an ontology