Arctic Region Supercomputing Center

Multisearch Grid Information Retrieval

Kylie McCormick

Mount Holyoke College ’08

Computer Science/

Theatre Arts

Greg Newby

ARSC Chief Scientist

Special Thanks to both Emily Teller and Chris Fallen.

ARSC

UAF UNIVERSITY OF ALASKA FAIRBANKS

- Information Retrieval (IR) research looks into finding desired information in unstructured data. The most popular use of information retrieval is in search engines.
- Grid technologies enable the use of distributed computing resources.
- Grid Information Retrieval (GIR) uses the tools and standards found in the grid community and applies them to distributed information retrieval.
- "Search Engines" are commonly monolithic – having one group that indexes, searches, and ranks the data.

Grid Information Retrieval

- However, monolithic search engines cannot index certain documents, like those that are password-protected.
- In GIR, it is possible to have user credentials to allow access to certain files, so that they can be searched.
- GIR relies on each system to provide indexing and searching for itself.
- Because of this, there are two major issues that are currently under research.

Grid Information Retrieval

- Any system that uses GIR will have to receive the result sets of other system's searches but will not necessarily be given any information that can help re-rank or re-weigh the document relevancy.
- Problem: When presented with sets of ranked and scored documents from multiple systems that each have different scoring methods, how can you accurately create a result set that presents the most relevant documents at the top?
- Research is currently looking into Merge Algorithms to solve this.

Merge Algorithms

Merge Algorithms

- Currently, ARSC is looking into three main merge algorithms:
  - Leap Of Faith – an algorithm that does not attempt to normalize scores, but simply accepts them as they are and re-ranks all documents based on their original scores
  - Rank Shuffle – an algorithm that assumes that a document's rank in its old system should be re-used in the final set of documents (so all documents from rank #1 will receive top ranking, and all documents from rank #2 will receive second ranking, etc.)
  - Naïve Merge – an algorithm that normalizes the scores of the documents in each set

- Systems that use GIR may have thousands of different servers that they can search.
- Problem: The bandwidth and wait associated with a large GIR system can make it unusable or undesirable compared to faster, monolithic systems like Google.
- In order to speed up searching in GIR, research is looking at restriction, or server-selection, algorithms that would limit the number of servers that the search will be run on.

Restriction Algorithms

- Currently, ARSC is looking into server-selection algorithms based on key terms obtained in Meta-Data.
- Simply put, the “most popular” terms in the server are compared to the query terms. Then the servers are each ranked according to their relevance to the query terms.
- Variations on this one particular algorithm include using ranked values for the terms to represent how frequently they appear in the server and simply using a boolean system of one and zero.

Restriction Algorithms

- Any GIR system could use both a restriction and a merge algorithm to improve its speed and result sets.
- Problem: How can we mix and match these algorithms without having unnecessary code and/or systems to test each algorithm pairing?
- ARSC has been working on Multisearch, a software that can be used to test multiple merge algorithms as well as multiple restriction algorithms in a mix-and-match fashion.

Combining Algorithms

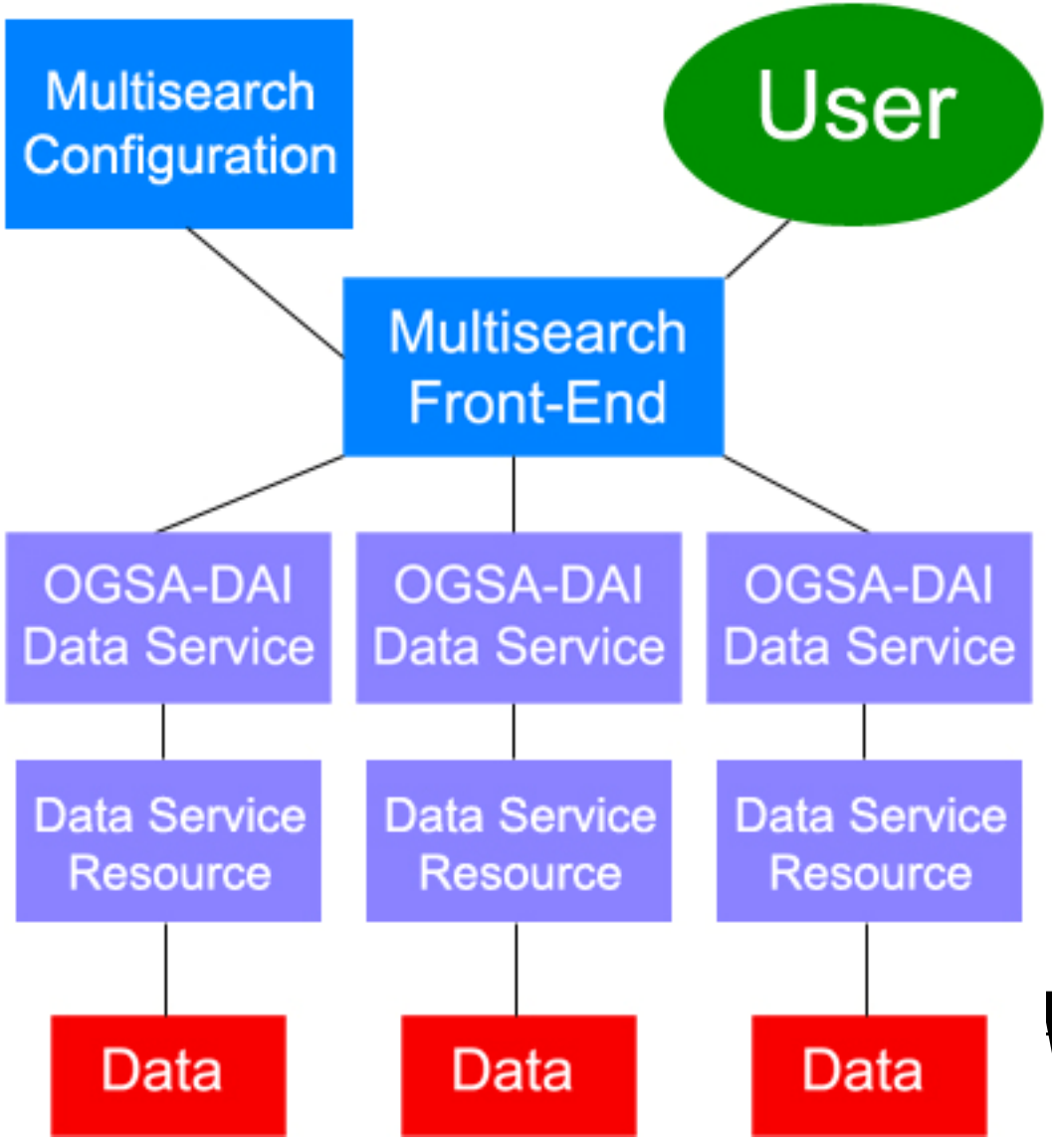
Multisearch Overview

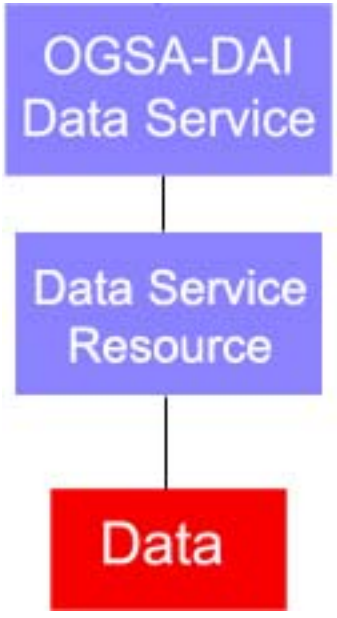
- Goal: Produce a package that can be used for research in GIR that is easy to use, allows mixing-and-matching for various algorithms, and is adaptable.

- Multisearch is flexible, allowing new additions and easy-to-edit XML code to configure the client.

- Multisearch is built off of other software, including OGSA-DAI WSI, Apache Tomcat, and Apache Lucene.

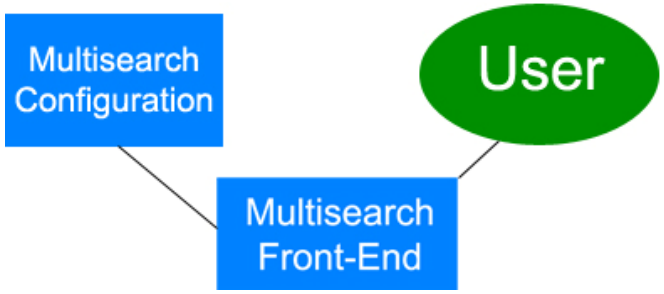
Multisearch


Multisearch Configuration
User
Multisearch Front-End
OGSA-DAI Data Service
OGSA-DAI Data Service
OGSA-DAI Data Service
Data Service Resource
Data Service Resource
Data Service Resource
Data
Data
Data



- OGSA-DAI Data Services allow resources, such as relational or databases, to be exposed in a grid environment.
- A Data Service can have multiple resources.
- The Data Service Resource provides data via a Data Resource Accessor, which reviews user credentials and gives access to the data stored on that data service.
- More information can be found @ http://www.ogsadai.uk.org/

Architecture Overview



- Multisearch comes with a Configuration object that extracts information found in an XML file.
- This XML defines
  - Which Algorithm is used to merge
  - Which Algorithm is used as a Server-Selection (if any)
  - What services and resources to search

Architecture Overview

- OGSA-DAI's Architecture aims to hide the various layers and heterogeneity of data on a grid system.
- Heterogeneity can provide difficulties with connections and querying. Multisearch uses OGSA-DAI for this very reason.
- Any new data resource – such as Egothor or Lemur or MySQL – can be added to Multisearch as long as it complies with OGSA-DAI standards and has a client toolkit (which generates XML for a query).
- This makes Multisearch more flexible than it has been in the past, and the user does not have to interact with the code to add new resources to search.

Architecture Benefits

- Multisearch can also have new algorithms added to it, so long as they comply with the standards in the user guide.
- Allowing users to add their own algorithms is essential to aiding new research in GIR.
- By having a configurable front-end client, the user does not need to have a data service in order to use Multisearch to run searches or do research.
- OGSA-DAI Standards are immensely beneficial to Multisearch, which can suffer from issues like varying requirements of XML query requests, which the client toolkit classes clear up.

Architecture Benefits


```
<configuration>
  <algorithm name="edu.arsc.multisearch.merge.LeapOfFaithMergeSet"/>
  <restriction name="edu.arsc.multisearch.restriction.LatterRestriction" ratio=".25"/>
  <services min="5" max="20">
    <node service="http://balto.arsc.alaska.edu:8080/axis/services/Domain4" name="Balto: Domain Set 1048">
      <resource id="domain4" clienttoolkitclass="edu.arsc.multisearch.LuceneSearchToolkit" secure="false" index="luc">
        <metaDataName>{http://ogsadai.org.uk/namespaces/2005/10/config}terms</metaDataName>
      </resource>
    </node>
  </services>
</configuration>
```

- By using XML, Multisearch can be configured to suit the objectives of the user.
- Each section of this XML can be modified as long as certain standards are met.

Syntax

<algorithm name="ClassName"/>

- Algorithm refers to the merge algorithm that is used to take the results from the various backend servers and combine them in a meaningful way.
- As long as a new merge algorithm extends edu.arsc.multisearch.merge.FinalSet and has a function called merge(), it can be added to Multisearch and this XML without modifying the code in Multisearch itself.
- There are currently three different algorithms in Multisearch: Naïve Merge, Rank Shuffle, and Leap of Faith.

Syntax

<restriction name="ClassName" ratio=".25" limit="10"/>

- Restriction is an optional field that refers to the server-selection algorithms that we have been testing this year.
- A new restriction algorithm must extend edu.arsc.multisearch.restriction.Restriction and must some how choose a top number of servers to be searched by Multisearch.
- In the XML, a ratio (how many servers from the servers provided) and a limit (how many servers maximum) must be given.
- Server-selection algorithms are used to improve performance by selecting servers that are "best" for the query, and thus eliminating possible noisy data from other backends and saving bandwidth.

Syntax

<services min="5" max="20"> … </services>

- The user can specify the minimum number of results a service needs to return to be valid, along with a maximum number to reduce bandwidth.
- Each service is added as a "node" with the URI, name of the server, and all of the resources in that service that are to be used.
- There needs to be at least one resource listed for a service, or else it will not have anything to search. If you are to update the XML file, you need to know the resource ID.

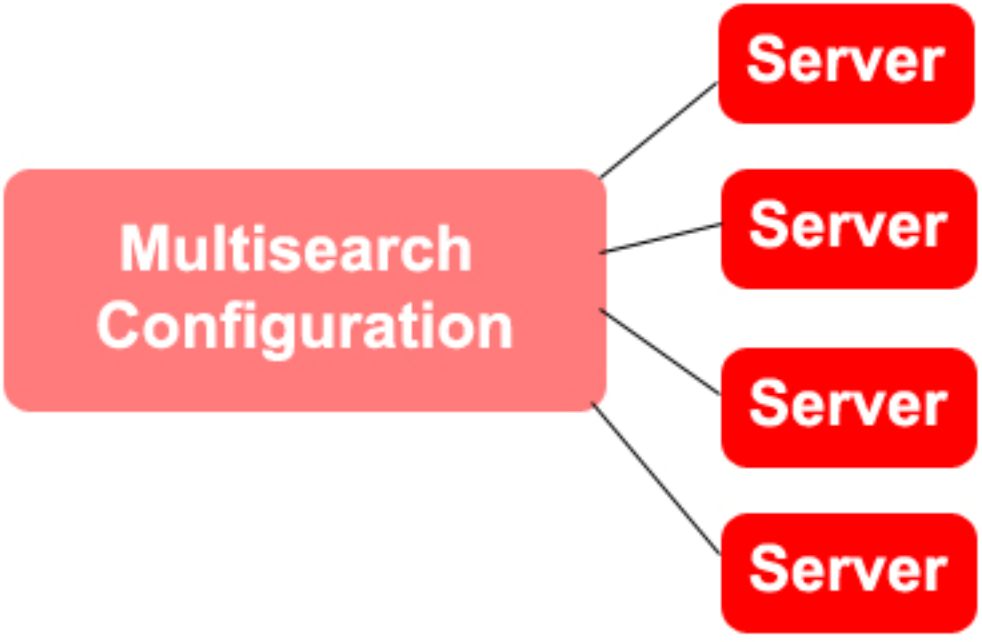
Syntax


```
<resource id="domain4" clienttoolkitclass="ClassName"
secure="false" index="luc">
    <metaDataName>
        {http://ogsadai.org.uk/namespaces/2005/10/config}terms
    </metaDataName>
</resource>
```

- The resource declaration is the longest—as you'll need to put in XML any additional things that the perform document might need. For example, Lucene requires an index to be passed to it, so for this particular service, we include an index.
- Meta Data is used to obtain information for the server-selection algorithm, and is based in OGSA-DAI's structure.

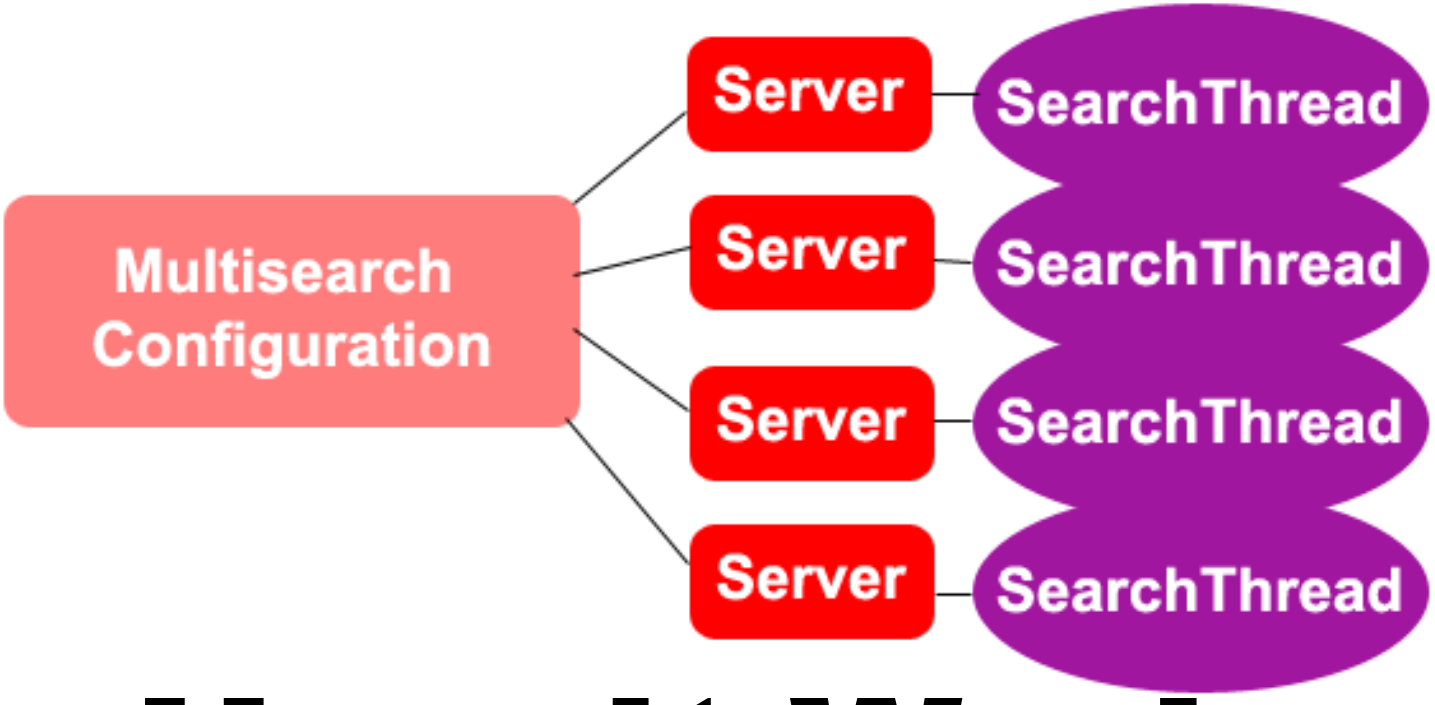
Syntax

1. Multisearch takes the XML Configuration file and makes a server object for each Data Service.



How it Works

2. The user submits a query string, and Multisearch creates a thread for each service/resource pair.

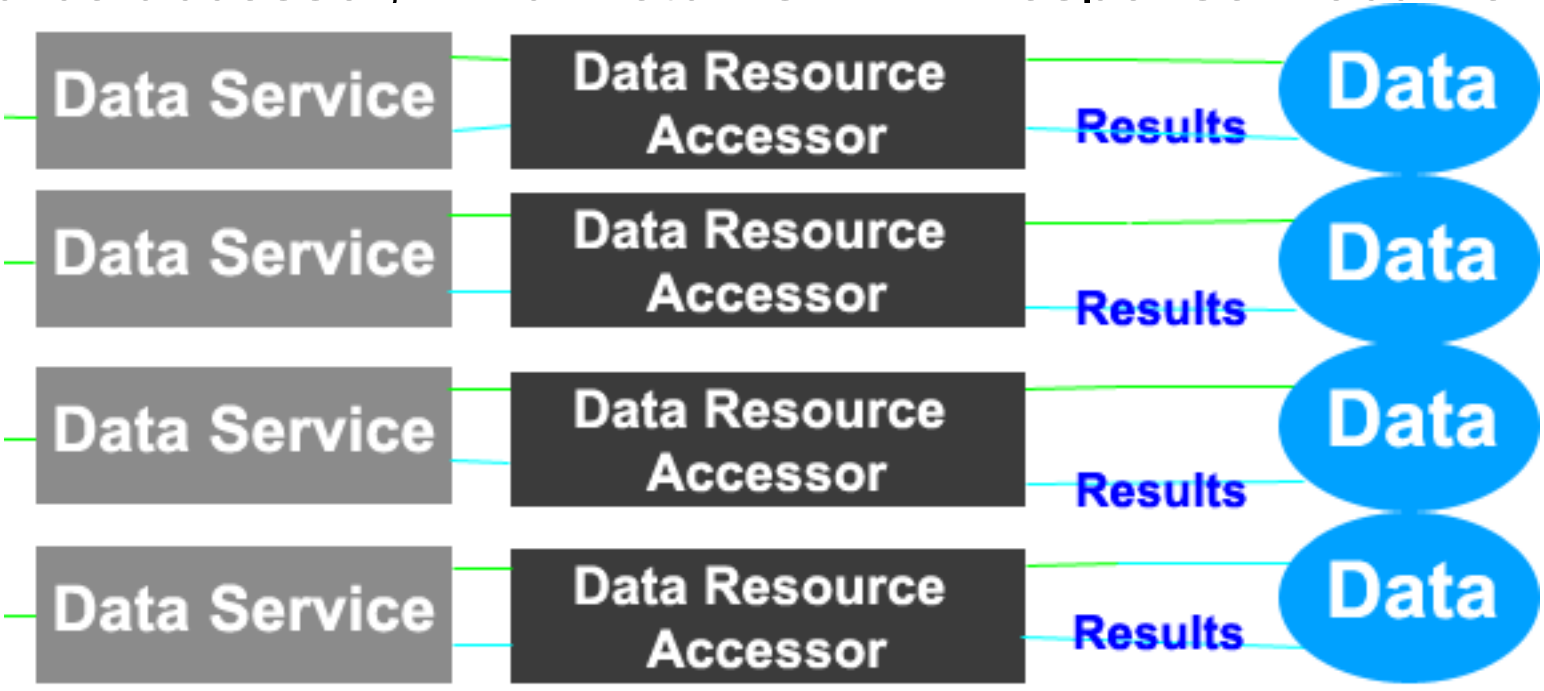


How it Works

3. The search thread uses the client toolkit to produce an XML Perform document which is sent to the data service.

How it Works

4. The data service passes the perform document to the data resource accessor, which returns XML Response Document.



How it Works

5. The response document is returned to the search thread, and the XML is converted into Document objects and passed back to Multisearch.

6. Each Document set is merged into a final list of documents by the algorithm provided in the XML document.

7. The results are presented to the user.

How it Works

```
snowy(mccormic) ~/merge/tomcat/webapps/edu/arsc/multisearch/WEB-INF/classes [204] > java edu.arsc.multisearch.Multisearch gov
ernment spending on educational improvements
Document Information: http://balto.arsc.alaska.edu:8080/domains/172/data/gov2.dsub.172/GX137-66-14417780
Title: Untitled ID: 1859 Server: Balto: Domain Set 172
Old Rank: 1 Old Score: 0.08386452
New Rank: 1 New Score: 0.08386452

Document Information: http://balto.arsc.alaska.edu:8080/domains/172/data/gov2.dsub.172/GX259-03-5542743
Title: Untitled ID: 6368 Server: Balto: Domain Set 172
Old Rank: 2 Old Score: 0.07801304
New Rank: 2 New Score: 0.07801304

Document Information: http://balto.arsc.alaska.edu:8080/domains/172/data/gov2.dsub.172/GX232-51-1428758
Title: Untitled ID: 163 Server: Balto: Domain Set 172
Old Rank: 3 Old Score: 0.07395286
New Rank: 3 New Score: 0.07395286

Document Information: http://balto.arsc.alaska.edu:8080/domains/1806/data/gov2.dsub.1806/GX024-57-0658251
Title: Untitled ID: 3204 Server: Balto: Domain Set 1806
Old Rank: 1 Old Score: 0.07011352
New Rank: 4 New Score: 0.07011352

Document Information: http://balto.arsc.alaska.edu:8080/domains/1806/data/gov2.dsub.1806/GX000-04-5838461
Title: Untitled ID: 3303 Server: Balto: Domain Set 1806
Old Rank: 2 Old Score: 0.051073782
New Rank: 5 New Score: 0.051073782
```

Command-Line

Arctic Region Supercomputing Center

Home

Multisearch Servlet Front-End

Welcome to the Arctic Region Supercomputing Center's Multisearch Frontend Servlet! Here you can test out the newest version of Multisearch from Summer 2007. The tools for this search include the following: Tomcat, Axis, OGSA-DAI, and Lucene.

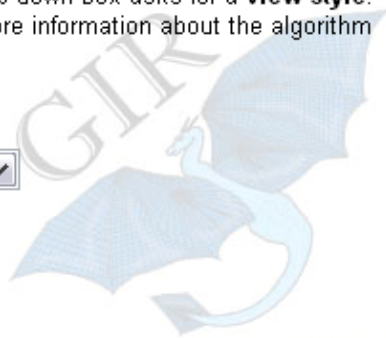
Multisearch searches over smaller backends instead of over a huge collection, as most search engines do.

The first drop-down box asks for a **merge algorithm**. This allows you to choose how the different sets are compared to one another and put together. The second drop-down box asks for a **restriction algorithm**, which selects various servers for your query and searches only those, to improve performance. The final drop-down box asks for a **view style**. Normal will simply show the results of the search; whereas, verbose will show you more information about the algorithm and the process.

Query: animal activity in the yee | Naive Merge | Latter | Verbose

Search Reset

Leap of Faith
Rank Shuffle
Naive Merge



Arctic Region Supercomputing Center
PO Box 756020, Fairbanks, AK 99775

http://snowy.arsc.alaska.edu:8080/edu/arsc/multisearch/MultisearchServlet

Any Questions?