CEWES MSRC Web-Linked Database Projects Developed by NPAC

Yuping Zhu, David E. Bernholdt
[yuzhu,bernhold]@npac.syr.edu

Abstract

This report summarizes two Focused Efforts with the common theme of coupling commercial database technology with the world-wide web to facilitate the presentation and management of large information spaces. The two projects are "Web-Linked Databases for Domain-Specific Information Repositories" and "Interfacing Databases and the Web: Management of Large WWW Sites Using Commercial Database Technology". Both projects utilize the Oracle7 relational database system, the Oracle Context Option and the Oracle Web Server. The services provided can be accessed using any standard web browser.

Web-Linked Database For Domain-Specific Information Repositories

Introduction

This project deals with searching out useful information from large amounts of information on World Wide Web. With the growth of Internet, the amount of information on the World Wide Web becomes increasingly large and it's hard for users to obtain what they want from the huge volume of information quickly, precisely and efficiently. This project provides an efficient way for user to locate specific information from the glut of information available on the web. It is a web search engine which centralizes and indexes information about grid generation technologies (as used in computational fluid dynamics, structural mechanics, and other fields). Grid generation expert Joe Thompson of Mississippi State University provided some of the initial information (URLs of major grid generation WWW sites) used to seed the search engine.

Components And Implementations

At the highest level, the Grid Search System consists of three parts: Robot, Database and Web Interface.

1. Robot

The robot is a gathe f four parts, as shown in Figure1.

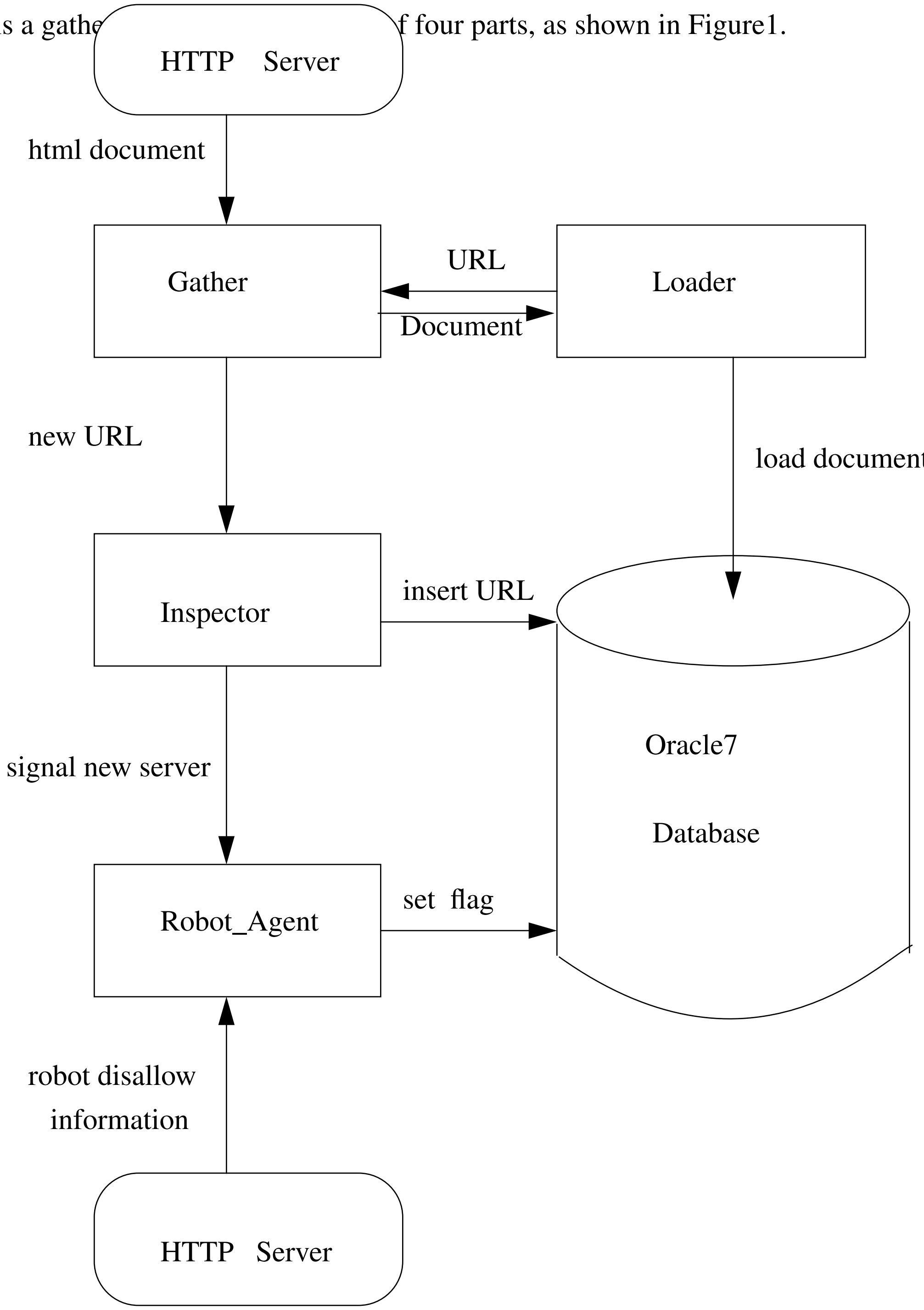


Figure 1. Grid Robot System

Given some starting URLs, the robot gathers HTML documents from remote web servers, following those URLs under the control of link-depth. Currently in the grid search engine, the link depth is 4.

1.1 Gather

The gather subsystem gets the URL to be gathered from the loader (the loader gets the URL from the URLs queue in database), then opens a network connection by creating TCP/IP socket to establish the communication with the specific HTTP web server, and gathers the HTML document from this web server using the HTTP protocol. Then it parses the document, extracting all links, document size, last update time, total references, total images and total forms, etc., and formats the document (to guarantee loading the document into database correctly). Then it appends all this information and the formatted document to a temporary file in order to load into database later. Any URLs found in the document are sent to the inspector subsytem by message communication. The gather subsystem is implemented in C.

1.2 Inspector

The inspector subsystem receives URLs from the gather subsystem and checks that they belong to the webspace of interes, and that it is not already in the database. If the URL is "valid", it is inserted into the links queue in the database to be gathered later, otherwise the URL is discarded. When a valid URL refers to a new web server, a signal is also sent to the Robot_Agent. The inspector subsystem is implemented in Pro*C.

1.3 Loader

The loader manages semaphore and message communication for the system. It passes URLs to be gathered from the links queue in the database to the gather subsystem, and receives information fmro the gather subsystem (including HTML documents) to be loaded into the database. This module is implemented in Pro*C.

1.4 Robot_Agent

This module has the simple task of checking the robot_disallowed file on new web servers It is implemented in Pro*C.

2. Database

The Database part can be divided into three parts: storage, index and search.

2.1 Storage

We use the Oracle 7 relational database system to store URLs, documents and other information. The layout of the table used to tore URLs and related data is shown in Table 1. Each URL occupies a row. Table 2 shows the layout of the table used to store documents and their associated data; each document occupies a row.

Field	Data Type
HOST_ID	NUMBER(6)
PROT	VARCHAR2(10)
HOST	VARCHAR2(60)
PATH	VARCHAR2(250)
OLDPATH	VARCHAR2(250)
FLAG	CHAR(1)
PRIORITY	NUMBER(2)
DEPTH	NUMBER(2)
FID	NUMBER(8)

Table 1. Layout of URL storage table.

Field	Data Type
FID	NUMBER(8) constraint pk_fid primary key,
STATUS	NUMBER(1),
HOST_ID	NUMBER(6),
PROT	VARCHAR2(10),
HOST1	VARCHAR2(60),
PATH	VARCHAR2(255),
TIME	DATE,
LTIME	DATE,
NO_REFS	NUMBER(5),
NO_HTML	NUMBER(5),
NO_IMGS	NUMBER(5),
S_FORM	NUMBER(1),
FSIZE	NUMBER(8),
TITLE	VARCHAR2(255),
SUBJECT	VARCHAR2(255),
TEXT	LONG

Table 2. Layout of document storage table.

2.2 Index

The index is an optional structure associated with tables and clusters. It provides a faster access path to table data. Properly used, indexes are a primary means of reducing disk I/O. Indexing is used at a fairly low level in the Oracle system, so that the presence or absence of an index does not require changes to any SQL statements, it merely effects the speed of execution.
Indexes are logically and physically independent of the data in the associated table. They can be created or dropped anytime without any effect on the base tables or other indexes. If an index is dropped, all applications continue to work. However, access to previously indexed data might be slower.

Once created, an index is automatically maintained and used by Oracle. Changes to data, such as inserting new rows, updating rows, or deleting rows, are automatically reflected in all relevant indices with no additional action by users. The presence of many indexes for a table decreases the performance of updates, deletes and inserts since the indexes associated with the table must also be updated.

In order to increase the performance of updating, deleting and inserting to document table, we adapt batch index mode to the text of a document. That is to say, we index the text of documents *after* loading all documents. For those non-text columns we use a real-time index, (e.g URL's id and document's id).

We use the Oracle ConText Option to create an index for the text of documents. First, we define a policy for the text column of document by calling CTX_DDL.CREATE_POLICY procedure and specify two required parameters: policy name and fully qualified column name for the policy. The example is as shown below:

```
execute ctx_ddl.create_policy(‘search’,’search.docs.text’)
```

Here, a policy named SEARCH is created for the TEXT column in the DOCS table owned by SEARCH.

After that, we create a text index for the text column by calling the CTX_DDL.CREATE_INDEX procedure. The example is as shown below:

```
execute ctx_ddl.create_index(‘search’)
```

2.3 Search

After a text index is created for a column, the ConText Server can process text queries for the column using the previously defined search policy.

The Oracle ConText Option is add-on component of the Oracle package. It provides powerful search, retrieval, and viewing capabilities for text stored in an Oracle database. These capabilities are implemented in Oracle using a ConText Server process, which complements the usual Oracle

Server processes, and a text index, created by ConText Option, for each database column in which text is stored.

We use PL/SQL package to implement the queries to the text of documents by calling CTX_QUERY.CONTAINS procedure. The example is as shown below:

```
ctx_query.contains('SEARCH', 'keyword','QUERY_TEMP',)
```

Here, SEARCH is the name of policy created for the text column. KEYWORD is the user's query expression. A query expression is a combination of query terms (words and phrases) and components, such as operators, that allow users to specify exactly which documents are to be retrieved.

There are several search options available for querying text, including:

a) Exact word or phrase

b) Associations
-- Stem of a word or phrase
-- Fuzzy match of a word or phrase which allows for misspelling
-- Words that sound similar to each other
-- Words specified as LIKE expressions
The expansion operators are shown in table 3.

Operator	**Symbol**	**Description**
STEM	$	Expands the list of words to include all words having the same stem or root word
FUZZY	?	Expands the list of words to include all words with similar spelling
SOUNDEX	!	Expands the list of words to include all words that sound the same

Table 3. The expression operators

c) Proximity
Search for words or phrases that appear close to one another in a document. Documents where the words or phrases are close together will score higher than those where the terms are farther apart.

d) Operators
Any of these options can be logically combined. Logical operators combine the term in a query expression. All single words and phrases may be combined with logical operators. When query term are combined, neither the number of spaces around the logical operator, nor the order of the query terms is significant, with exception of the MINUS operator(i.e., 'A minus B ' is different

from ‘ B minus A ‘). All operators are shown in table 4:

Operator	Symbol	Equivalent
AND	&	and
OR	|	or
ACCUMULA	,	accum
MINUS	-	minus

table 4. Logical operators and meaning

CTX_TEMP ***(is this ‘QUERY_TEMP’ in the ctx_query.contains invocation above? If not, what is it? I don’t see CTX_TEMP anywhere else in this document)*** is the results table created before the query is performed. The result table is always called the “hitlist” table, and it must have the structure illustrated by the Table 5.

Field	Data Type
TEXTKEY	VARCHAR2(64)
SCORE	NUMBER
CONID	NUMBER

Table 5. The architecture to the query results and relative data type

Using the hitlist, the grid search system displays the document list page by page, including title, size, last update time and subject. If the user wants to a particular document, he can click on the document’s entry in the list using mouse and the whole document will be displayed with the key-words are highlighted.
We implemented viewing document in Pro*C by calling the procedure CTX_QUERY.HIGHLIGHT. This procedure fetches the document, parses the query, identifies the matching terms and insert results into the HIGHLIGHT_TEMP table.
The example is shown in the Table 6.

Field	Data Type
ID	NUMBER
OFFSET	NUMBER
LENGTH	NUMBER
STRENGTH	NUMBER

Table 6. HIGHLIGHT_TEMP table payout

Using the offset and length of the keywords in the document, we form the highlighted version of the HTML document.

3. Web Interface

In the grid search system, we use Oracle WebServer as our search engine's server. The Oracle WebServer is a HyperText Transfer Protocol (HTTP) Internet Server with unprecedented database integration and a powerful development environment. When the WebServer receives a URL from a browser located either on the World Wide Web or on a local network, it draws on information from the database and the operating system's (OS) file system as necessary to respond to the request. The file system can be used for static (hardcoded) Web pages, or for CGI scripts that do not access the database, and the database is used for Web pages that are generated at runtime using "live" data.

One of the important components of the Oracle WebServer is the Web Listener. It receives a URL from a web browser and sends back the appropriate output. When the Web Listener receives a URL, it determines whether the request requires the use of a Service, accessed through the Web Request Broker (WRB), a program, accessed through the CGI interface, or whether access to the file system of the machine on which the Listener resides is sufficient. If WRB access is required, the Listener passes the request to the WRB Dispatcher for processing; then it returns to the task of listening for more incoming HTTP requests.

The Dispatcher maintains communication with a pool of processes called WRB Executable Engines (WRBXs). Each WRBX interfaces to a back-end application using the WRB API, These applications are called WRB cartridges. The PL/SQL agent is one of such cartridges. A WRBX that is to use the PL/SQL agent immediately connects to the database when it is created and waits for a request to come in. This enables requests to proceed much more quickly.

The PL/SQL agent executes application code written in PL/SQL and returns the output in HTML form for the Web Listener to output as a Web page. When a URL is received for the PL/SQL agent, it contains a Database Connection Descriptor (DCD). The DCD determines both the database access privileges the PL/SQL agent has when executing this request and the schema (portion of the database) that it accesses. PL/SQL procedures are stored in the database. The PL/SQL agent invokes these by issuing commands to the database, which then performs the actual execution and sends the output and status messages back to the PL/SQL agent.

Based on Oracle WebServer, we provide the user with a web interface for entering the query string. The user can type any desired search expression . The main Web Interface is shown in Figure 1. After accepting the user's query string, the interface calls the PL/SQL package through the PL/SQL agent, and returns the hitlist. The interface is illustrated in Figure 2. Then the user can click on the document hitlist to view a particular document with the keywords highlighted. This is illustrated in Figure 3.

WWW Sites Management System Using Commercial Database Technology

Introduction

The purpose of this project is to couple commercial relational database technology with the World Wide Web to make it easier to effectively manage information on the WWW site of a large organization such as the CEWES MSRC, including obtaining overview information, such as web page distribution, last update time, number of images and links, as well as finding “dead links” and “orphaned files”.

Components and Implementation

This prototype Web Site Management System (WSMS) includes a search engine, structural overview of the site, as well as search capabilities for the structural information. Dead link identification comes naturally from the structural information, and orphan file identification is implemented using a Perl script to get around the fact that the prototype WSMS is not actually installed on the site’s web server system.

1. Search Engine

The search engine component of the WSMS is essentially the same as that used for the grid search engine. Of course it only gathers and indexes the document from one specific web server, and it does not use link-depth control to limit gathering of the site. The implementation is the same as that of grid search system and the interface is illustrated in Figure 4.

2. Structural Summary

This component provides the web site summary information for the webmaster. In order to improve the performance, whenever the database is updated, it is generated from database and stored as static HTML file. It provides the web information as shown as below:

-- Total pages: The total number of HTML documents on the web site.
-- Total linked Images: The total number of images linked by documents on the web site.
-- Dead links: The total number of dead links within the web site.
-- Total size: The total size of all documents on the web site.
-- Longest size: The size of the longest document on the web site.
-- Shortest size: The size of the shortest document on the web site.

This componenet is implemented using PL/SQL and the interface is illustrated in Figure 5.

3. Web page distribution structure

For the webmaster, if the web site is big enough, it’s always hard to get a handle on how the documents are distributed on the web site. This component provides a mechanism for the webmaster to get such information. It begins with the root directory of web server and follows the all subdirectory to display the documents and their structural information. It may be different from the real file system directory structure on web site, because it's based on the information in the database, which only stores the HTML documents obtained by accessing the site's HTTP server. It is implemented using PL/SQL and the interface is illustrated in Figure 6.

4. Web structure information search

Sometimes the webmaster may want to search the information on the web site according to the document's structure instead of content (keyword search). An interface is provided to allow the structural information described above to be searched. Possible queries include: size, last update time, or number of links or images. This functionality is also implemented using PL/SQL and the interface is illustrated in Figure 7.

5. Orphan file finder

There are some documents on the web site which can not be referenced by any other document, we call these documents "orphan files". The orphan files can not be gathered by the robot, so they are not stored in database; they only can be found on web site's filesystem. Our orphan file finder can be divided into two parts. One part is located on the database side, and generates the whole document list from database and stores the list on the database server. It's implemented using Pro*C. The other part is a Perl script located on the web site host system. It gets the document list generated by the first part from the remote HTTP server and compares it with the files in the real file system. Files present on the web site filesystem, but which are never referred to by other documents in the site are flagged as orphans. It is important to note that "orphan files" may in fact be correct in some cases, such as documents referred to by other web sites, or documents intended for limited distribution.

Conclusions and Future Vision

These two projects demonstrate the ease and power of coupling commercial database technologies with the world-wide web in working with large information spaces.

The grid search system is representative of focused, domain-specific search engines that can be easily created using this approach. It allows DoD researchers interested in grids and grid generation to more easily and efficient extract relevant information from the flood of information now available on the Internet. Search engines for other domains are planned, and there is ample scope to improve the algorithms used to filter information as it is being gathered into the database.

The prototype web site management system uses the same basic facilities as for the search engine to provide a tool to ease the management of a large, rapidly changing web site with many individuals contributing to the content. When the full system is installed on-site at CEWES, it will be possible to serve the web site from the WSMS. At that point, content providers will be able to check documents into and out of the WSMS, and a security model will provide for concepts of ownership, release approval, and staging of documents from private (pre-release) webspace to public webspace, in addition to the types of support the system already provides to the webmaster.

While useful in their own right, these projects are also meant to serve as more general exemplars of the advantages of coupling databases with the WWW. The HPCC community historically has not made much use of commercial database technology, despite the large volumes of complex data often arising in their work. We believe that the ability to put an easily accessible web-based interface onto large databases goes a long way towards lowering the historical barrier to the use of databases in HPCC, and we will use the search engine and WSMS to promote increased use of these technologies by DoD researchers.

References

1. Oracle Corp., Oracle7 Server, Release 7.3 Reference, 1996
2. Oracle Corp., Oracle ConText Option Application Developer's Guide, Release 1.1, 1996
3. Oracle Corp., Oracle WebServer User's Guide, 1996

Figure 1


Netscape: GRID Search Engine
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
Location: http://osprey7.npac.syr.edu:1963/grid/
What's New? What's Cool? Destinations Net Search People Software
Welcome
To
GRID Search Engine
Keyword Search: grid
Submit Help
Search Type: Keyword Max Depth: 3 Country: ALL
GRID Search Engine is developed at Northeast Parallel Architectures Center at Syracuse University.
Send comments to Yuping Zhu

Figure 2

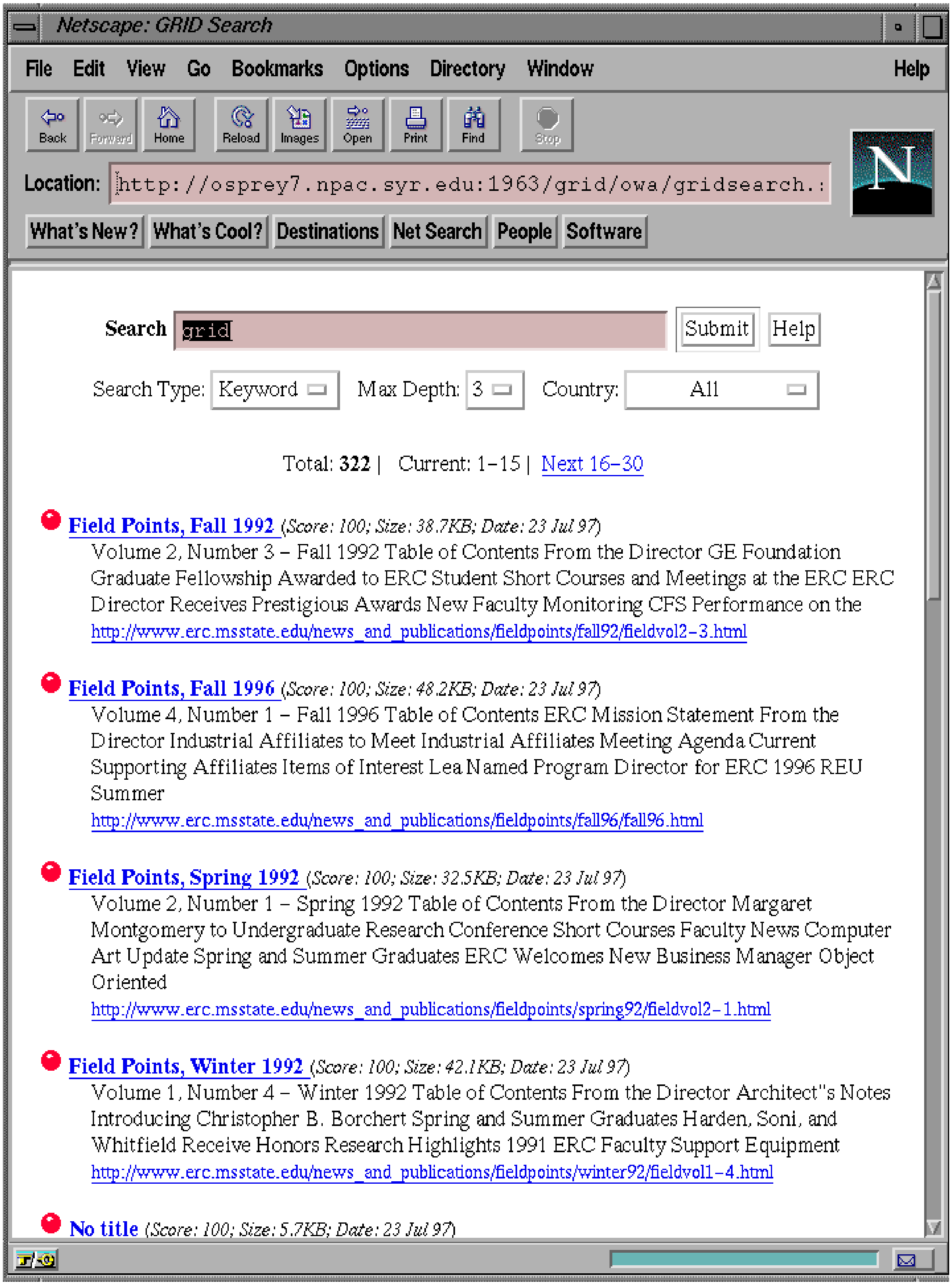

Netscape: GRID Search
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
Location: http://osprey7.npac.syr.edu:1963/grid/owa/gridsearch.:
What's New? What's Cool? Destinations Net Search People Software
Search grid Submit Help
Search Type: Keyword Max Depth: 3 Country: All
Total: 322 | Current: 1-15 | Next 16-30
Field Points, Fall 1992 (Score: 100; Size: 38.7KB; Date: 23 Jul 97)
Volume 2, Number 3 – Fall 1992 Table of Contents From the Director GE Foundation Graduate Fellowship Awarded to ERC Student Short Courses and Meetings at the ERC ERC Director Receives Prestigious Awards New Faculty Monitoring CFS Performance on the
http://www.erc.msstate.edu/news_and_publications/fieldpoints/fall92/fieldvol2-3.html
Field Points, Fall 1996 (Score: 100; Size: 48.2KB; Date: 23 Jul 97)
Volume 4, Number 1 – Fall 1996 Table of Contents ERC Mission Statement From the Director Industrial Affiliates to Meet Industrial Affiliates Meeting Agenda Current Supporting Affiliates Items of Interest Lea Named Program Director for ERC 1996 REU Summer
http://www.erc.msstate.edu/news_and_publications/fieldpoints/fall96/fall96.html
Field Points, Spring 1992 (Score: 100; Size: 32.5KB; Date: 23 Jul 97)
Volume 2, Number 1 – Spring 1992 Table of Contents From the Director Margaret Montgomery to Undergraduate Research Conference Short Courses Faculty News Computer Art Update Spring and Summer Graduates ERC Welcomes New Business Manager Object Oriented
http://www.erc.msstate.edu/news_and_publications/fieldpoints/spring92/fieldvol2-1.html
Field Points, Winter 1992 (Score: 100; Size: 42.1KB; Date: 23 Jul 97)
Volume 1, Number 4 – Winter 1992 Table of Contents From the Director Architect''s Notes Introducing Christopher B. Borchert Spring and Summer Graduates Harden, Soni, and Whitfield Receive Honors Research Highlights 1991 ERC Faculty Support Equipment
http://www.erc.msstate.edu/news_and_publications/fieldpoints/winter92/fieldvol1-4.html
No title (Score: 100; Size: 5.7KB; Date: 23 Jul 97)

Figure 3

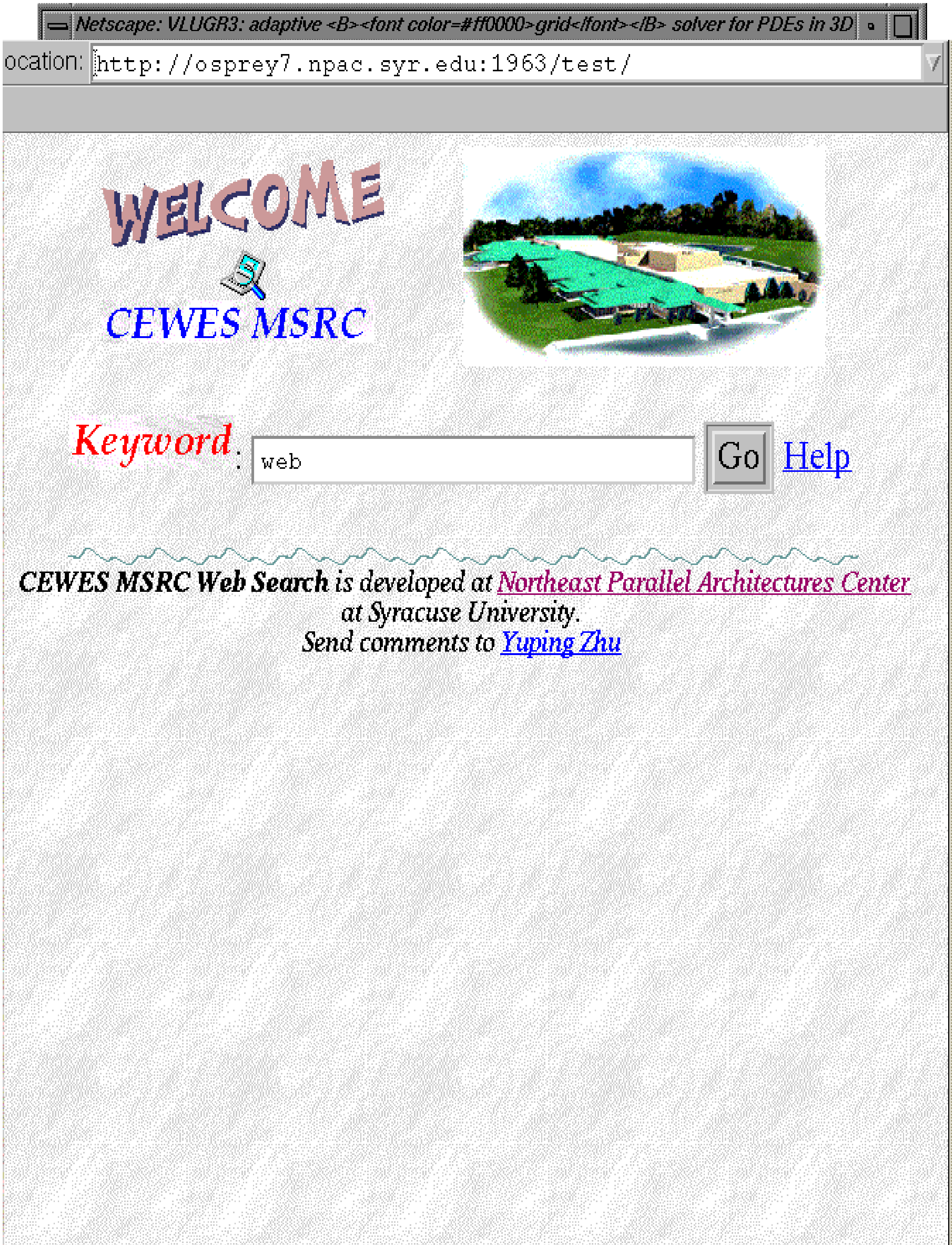

Netscape: VLUGR3: adaptive <B>grid</B> solver for PDEs in 3D
ocation: http://osprey7.npac.syr.edu:1963/test/
WELCOME
CEWES MSRC
Keyword:
web
Go
Help
CEWES MSRC Web Search is developed at Northeast Parallel Architectures Center
at Syracuse University.
Send comments to Yuping Zhu

Figure 4

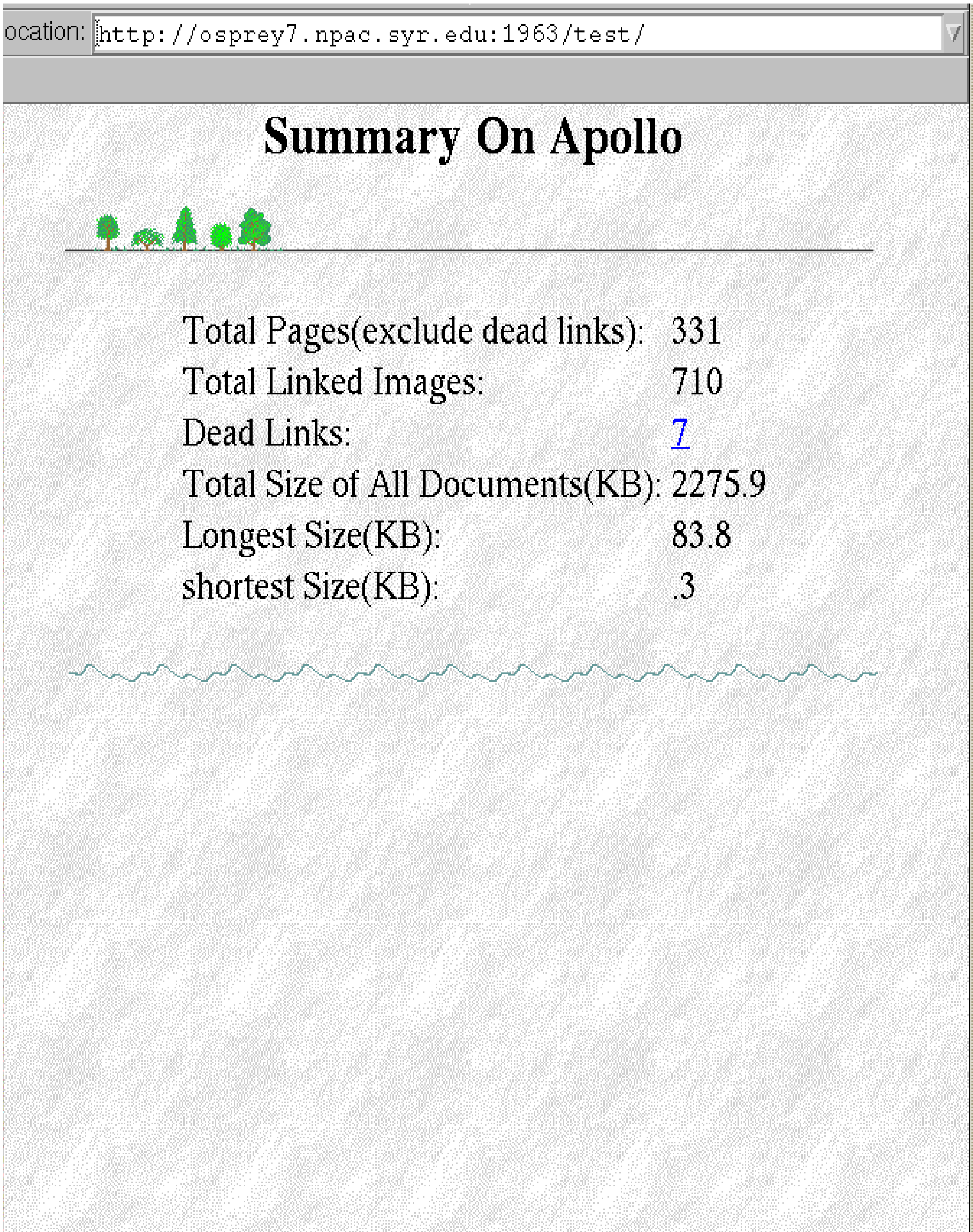
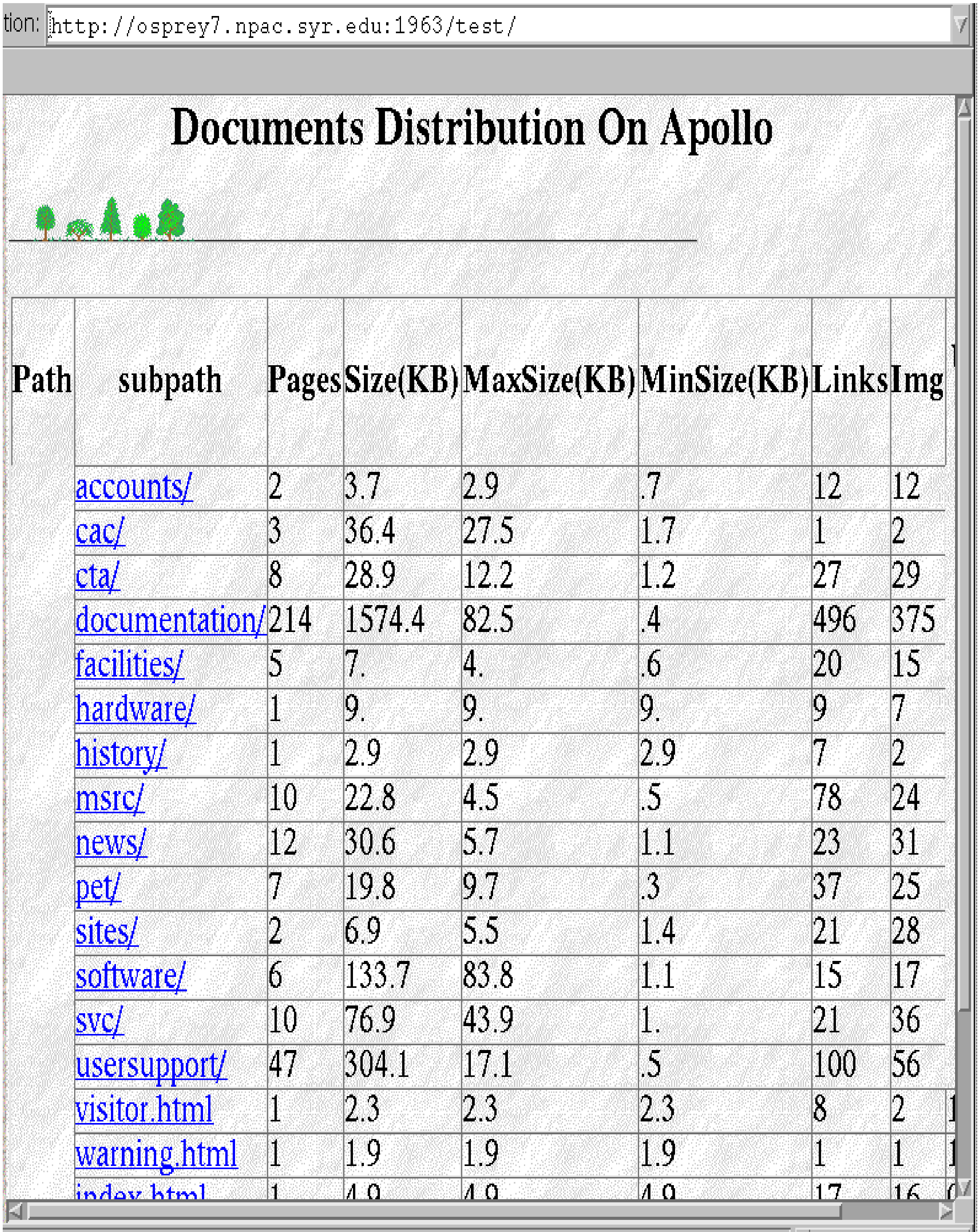

ocation: http://osprey7.npac.syr.edu:1963/test/
Summary On Apollo
Total Pages(exclude dead links): 331
Total Linked Images: 710
Dead Links: 7
Total Size of All Documents(KB): 2275.9
Longest Size(KB): 83.8
shortest Size(KB): .3

Figure 5



tion: http://osprey7.npac.syr.edu:1963/test/

Documents Distribution On Apollo

Path	subpath	Pages	Size(KB)	MaxSize(KB)	MinSize(KB)	Links	Img
	accounts/	2	3.7	2.9	.7	12	12
	cac/	3	36.4	27.5	1.7	1	2
	cta/	8	28.9	12.2	1.2	27	29
	documentation/	214	1574.4	82.5	.4	496	375
	facilities/	5	7.	4.	.6	20	15
	hardware/	1	9.	9.	9.	9	7
	history/	1	2.9	2.9	2.9	7	2
	msrc/	10	22.8	4.5	.5	78	24
	news/	12	30.6	5.7	1.1	23	31
	pet/	7	19.8	9.7	.3	37	25
	sites/	2	6.9	5.5	1.4	21	28
	software/	6	133.7	83.8	1.1	15	17
	svc/	10	76.9	43.9	1.	21	36
	usersupport/	47	304.1	17.1	.5	100	56
	visitor.html	1	2.3	2.3	2.3	8	2
	warning.html	1	1.9	1.9	1.9	1	1
	index.html	1	4.9	4.9	4.9	17	16

Figure 6

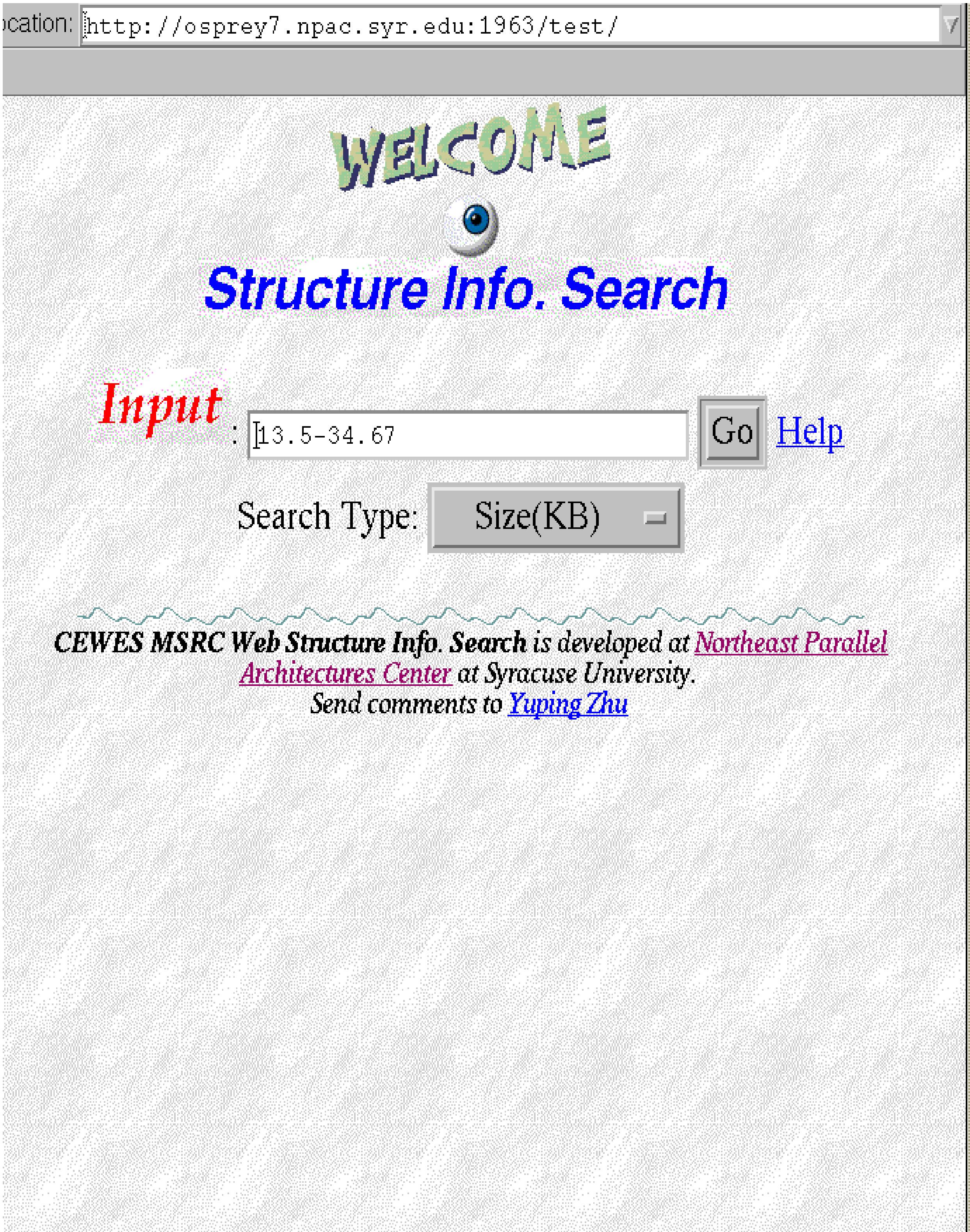

ocation: http://osprey7.npac.syr.edu:1963/test/
WELCOME
Structure Info. Search
Input : 13.5-34.67
Go
Help
Search Type: Size(KB)
CEWES MSRC Web Structure Info. Search is developed at Northeast Parallel Architectures Center at Syracuse University.
Send comments to Yuping Zhu

Figure 7