AUTOMATED ARTICLE GENERATION USING THE WEB

A Writing Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Gaurang Patel

December 2009

SAN JOSÉ STATE UNIVERSITY

The Undersigned Writing Project Committee Approves the Writing Project Titled

AUTOMATED ARTICE GENERATION USING THE WEB

by Gaurang Patel

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

Dr. Chris Pollett, Department of Computer Science	12/17/2009
Dr. Cay Horstmann, Department of Computer Science	12/17/2009
Dr. Mark Stamp, Department of Computer Science	12/17/2009

ABSTRACT

AUTOMATED ARTICE GENERATION USING THE WEB

by Gaurang Patel

An article generation application is an intelligent mining engine that looks for web content, then combines and organizes this content in a meaningful way to generate an article. This contrasts with a search engine which generates a list of links to pages containing keywords. This writing project is about such an article generation tool. Our tool generates articles on the topic entered by the user using information available on the web. The articles have well defined sections, each talking about different aspect of the topic.

ACKNOWLEDGEMENTS

I am grateful to my project advisor Dr. Chris Pollett for his guidance throughout year. I would also like to thank Dr. Cay Horstmann and Dr. Mark Stamp for their time and feedback. Mr. Ayyappan Arasu deserves a special thanks for answering my concerns at various stages during the coding of my project. I am also grateful to the developers and users of both the Carrot^2 and the Nutch for their responses to my questions on various discussion forums.

Table of Contents

List of Figures

List of Tables

1. Introduction

Often when trying to find information on the web, one makes use of a search engine. A query to such a search engine consists of a list of keywords. The search engine responds with a web page containing links to pages with that keyword. It does not combine these results into one resource like an article. A user often has to visit a number of pages to find what he wants. For this project, we created an automated article generation engine which can produce articles out of these links. The goal of the project is to be able to produce articles with relevant sections of text in it. An article is a meaningful collection of sections, each of which talks about different aspect of the topic. Sections are thematic categories for the entered topic. Our Article Generation Engine generates article sections with text information. Data in other formats, for example, images, links, etc, is not considered in article generation.

Our system can be contrasted with other sources of articles on the web. Often such sites provide static articles (e.g. Wikipedia) which are user contributed. For such static articles, there might be accuracy and bias issues. One of the goals of this project is to have the Article Generation Engine produce as accurate information as possible. Clustering and text summarizing techniques are used to mine the information into sections and to derive the gist of each of the sections respectively.

The project is mainly divided into two parts. The first deliverable is to develop and test each system module individually. It includes building the Crawler/Indexer, the Clustering Engine and the Summarizer. The second part of the project is combining these parts to have the final Article Generation Engine ready and capable of generating articles. This includes developing a website, integrating the basic modules with the website and implementing noise reduction techniques.

The project report explains how the Article Generation Engine was developed. It includes details on each system module as well as how these modules were integrated. It also discusses noise reduction techniques. It is organized as follows: Sections 3, 4 and 5 talk about each of three basic modules in detail. These sections include how each system module was developed. Section 7 mentions steps for integrating these modules into our Article Generation System. Website development and noise reduction techniques are discussed in Section 6 and 8 respectively. Section 10 analyzes the article generation results. Section 11 concludes the paper.

2. System Architecture

Our Article Generation Engine is a complex system as it has modules that are developed in different programming languages. These programming languages include C, Java and PHP. Thus, one part of making these modules to communicate was to pass data among functions in these different languages.

2.1. System Modules

The Article Generation System is comprised of three core modules:

The Crawler/Indexer/Searcher

The Article Generation Engine is dynamic in the sense that it fetches information from the web to generate an article. The indexer and the crawler behind it play an important role in the efficiency and performance of the system. The purpose of this module is to be able to obtain search results on a given query. This module was built in a way that it could make use of different open source technologies for performing web search. One of these that we considered was the Nutch search engine. The other was the Google search API.

The Clustering Engine

An article is a collection of well-organized, relevant and informative paragraphs/sections. The clustering module of this project was used to determine which web pages on the entered topic are related and might be useful to create such sections. This module was built on top of the Carrot^2 clustering engine [1]. The clustering engine, after receiving search results, organizes them into meaningful topics and assigns certain web pages to each cluster.

The Text Summarizer

Clustered documents need to be summarized to generate appropriate content to be displayed in the relevant section. The text summarizer module is responsible for this step. This module was designed so that it could use different open source text summary engines. In particular, experiments on OTS (Open Text Summarizer) and Great Summary were carried out in order to obtain sample summaries. A Carrot^2 plug-in was also developed for summarizing a page. Section 5 talks about the summarizing module in detail.

2.2. Architecture

Figure 2.2.1 illustrates the architecture of the project. After each of the previously discussed modules had its turns operating on the data, noise reduction techniques were used to tune the article quality.

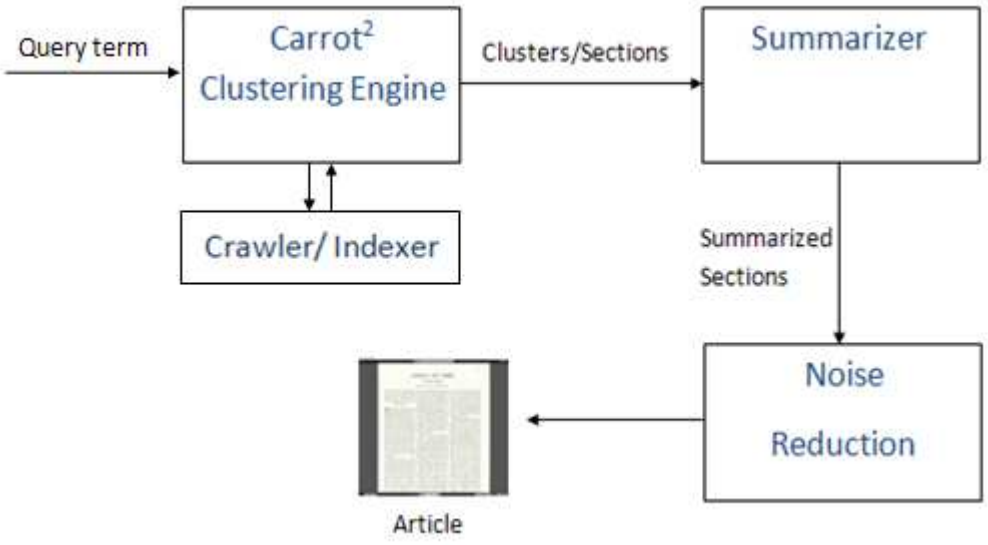

Figure 2.2.1: System architecture

3. Crawler/Indexer/Searcher

A couple of crawler-indexer-searchers were considered for this project.

3.1. Nutch

Nutch is an open source search engine/crawler. It builds on top of Lucene (a text search engine library). Nutch is written in Java. We next briefly describe how Nutch can be deployed and how it was used with our system.

3.1.1 Sample Nutch Crawl and Search

Crawling

Nutch configuration consists of the steps of setting the agent name and domain name, creating a URL file and creating a crawl directory.

Nutch supports command lines for crawling:

```
$ bin/nutch crawl urls -dir crawl -depth 3 -topN 50
```

Search the crawled results.

- Enable clustering plug-in in the nutch-site.xml file by adding a property.
- Deploy the web application that comes with Nutch to the Tomcat application server and run it in a browser.

In this sample run, the "http://www.yahoo.com" domain was crawled to the depth of five levels starting from the URL "http://sports.yahoo.com", fetching top 1000 results at each level.

The crawl command is:

```
$ bin/nutch crawl urls -dir crawl.sports.yahoo-5-1000 -depth 5 -topN 1000
```

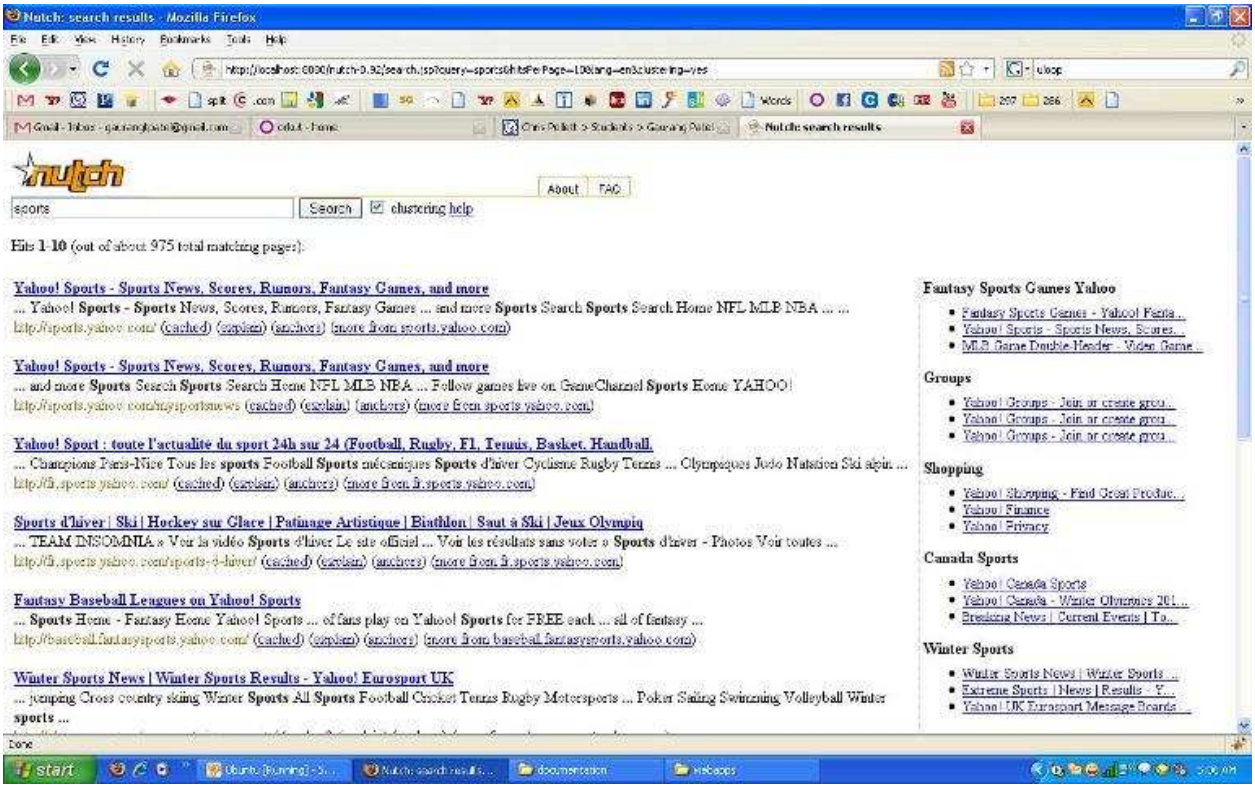

Figure 3.1.1: Search and clustering results using Nutch and Carrot^2

The left panel on the page in Figure 3.1.1 shows the search results for the query "sports". Groups on the right panel of the page are the clusters found in these search results, if the "clustering help" check box is selected. It uses the Carrot^2 clustering plug-in that comes with Nutch.

3.1.2 Crawling the Whole Web

The Article Generation Engine requires the whole web to be crawled, indexed and ready to be used with the clustering module. Whole web crawling requires totally different steps to be followed. The `crawldb` is injected with a list of URLs, crawl sections are generated, and crawling is applied. Here we use the DMOZ open directory [18] for injecting `crawldb`. The DMOZ directory has about 4.5M URLs. Observations during a whole web crawl came out with memory

and processing efficiency concerns. Nutch spends 45 minutes to crawl 16k URLs. This equates to the time of 19 days to crawl 10M URLs, which is still less than the size of the whole web.

3.2. Google Search Results

Another way we obtained search results for our system was to use a Google API. Carrot^2 comes with a source library named `GoogleDocuments`, which automatically searches for a term on Google and returns the results. These results can then be used for Carrot^2 core libraries to form the article clusters.

4. Carrot^2 Clustering Engine

The clustering engine is responsible for generating the article sections. It can be thought as the first phase in the article generation. As discussed earlier, Carrot^2 is used as the clustering engine for this project. Carrot^2 is an open source search results clustering engine. It can organize small collections of documents into thematic categories [1]. Clustering plays an important role in the article generation.

Challenges

Carrot^2 is a large project. The stable branch of the project has a total of 65 sub-projects/plug-ins and 700 Java files. Exploring and modifying Carrot^2 was difficult, but the Eclipse IDE made it easier. Eclipse's project explorer made it easy to explore through the Carrot^2 core libraries and Carrot^2 examples. Moreover, Carrot^2 is written in Java. Integrating the Carrot^2 clustering algorithm with the website, which was written in PHP, was also a challenge.

4.1. Exploring the Carrot^2

The GNU tarball can be used in the Eclipse IDE to create projects in an Eclipse workspace. Source code can then be modified and various scenarios can be tested within Eclipse. Moreover, Carrot^2 provides a Tomcat deployable web application. Section 3.1 discusses more on this web application. For this project, we used the version 3.0.1 of Carrot^2.

The Carrot^2 document clustering workbench is a desktop application that can be used to run sample clustering processes and explore clustered results visually. It can be useful to understand the scope and functionalities of Carrot^2. Figure 4.1.1 illustrates a workbench run for the query "India". This particular run uses the search results from a Yahoo source and the Lingo

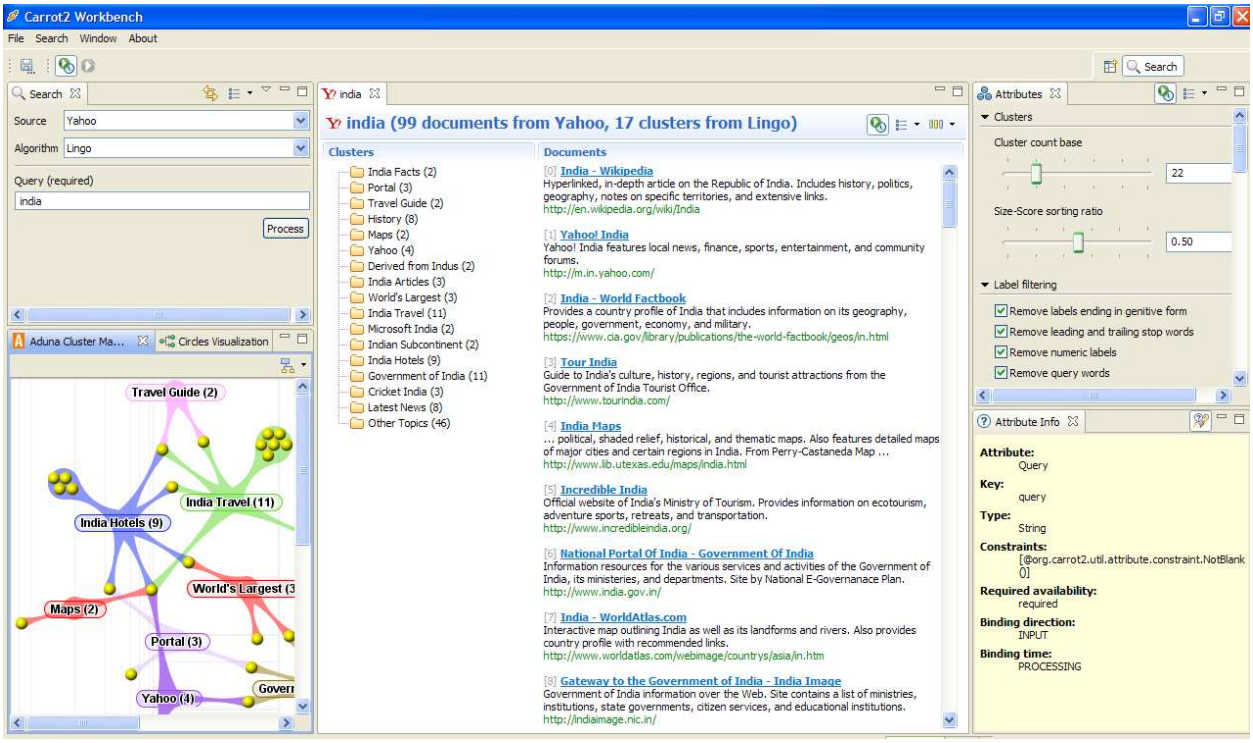

Figure 4.1.1: Carrot^2 workbench run for query "India" using the Lingo algorithm and a Yahoo source

algorithm as the clustering algorithm. The number and size of the clusters can be tuned using the panel on the right hand side of the tool. The bottom-left panel visually represents clusters and their relations.

Testing Carrot^2 through the workbench helped us learn several things: The available list of algorithms in Carrot^2, the available indexers/search engines that can be used to provide search results to Carrot^2, and so on.

4.2. Clustering Sample Run

We modified the program ClusteringDataFromDocumentSources.java program to test Carrot^2 where search results were returned from Google.

Following is the code snippet from ClusteringDataFromDocumentSources.java

```
SimpleController controller;
Map<String, Object> attributes;
ProcessingResult result;

controller = new SimpleController();
attributes = new HashMap<String, Object>();

attributes.put(AttributeNames.QUERY, args[0]);
attributes.put(AttributeNames.RESULTS, 10000);

result = controller.process(attributes,
        GoogleDocumentSource.class, LingoClusteringAlgorithm.class);

ExampleUtils.displayResultsPHPUnderstandable(result);
```

Figure 4.2.1: Sample clustering code snippet

Briefly, the snippet above works as follows: The `SimpleController` class defines the lifecycle of a Carrot[2] processing component. The life cycle governs how the controller instances are initialized and disposed of and how the processing operates. The attributes variable holds a list of parameters needed during the clustering process. The parameters include query string, maximum number of results to fetch, etc. The argument GoogleDocumentSource.class in the `SimpleController::process()` method indicates that the Google search results are being used for clustering. The second parameter, LingoClusteringAlgorithm.class, indicates that the Lingo clustering algorithm will be used of the three available clustering algorithms in Carrot[2]. The query string is provided as an argument to this Java program. The `ExampleUtils` class provides methods to output the clustering results to the standard output.

Output

```
2009-12-02 14:49:38,408 INFO  org.carrot2.clustering.lingo.LingoClusteringAlgorithm: Native BLAS r
2009-12-02 14:49:38,531 DEBUG org.carrot2.util.httpclient.HttpUtils: GET: http://ajax.googleapis.c
2009-12-02 14:49:38,542 DEBUG org.carrot2.util.httpclient.HttpUtils: GET: http://ajax.googleapis.c
2009-12-02 14:49:38,543 DEBUG org.carrot2.util.httpclient.HttpUtils: GET: http://ajax.googleapis.c
2009-12-02 14:49:38,544 DEBUG org.carrot2.util.httpclient.HttpUtils: GET: http://ajax.googleapis.c
Collected 32 documents

[ 0] India - Wikipedia, the free encyclopedia
     http://en.wikipedia.org/wiki/India

[ 1] CIA - The World Factbook -- India
     https://www.cia.gov/library/publications/the-world-factbook/geos/in.html

[ 2] Incredibleindia.org | Home page
     http://www.incredibleindia.org/

[ 3] Home: National Portal of India
     http://india.gov.in/

[ 4] India Travel Information and Travel Guide - Lonely Planet
     http://www.lonelyplanet.com/india

[ 5] Welcome to India - for Tourism, Travel, Visit and to Explore
     http://www.tourindia.com/

[ 6] India (11/09)
     http://www.state.gov/r/pa/ei/bgn/3454.htm

[ 7] Embassy of India - Washington DC
     http://www.indianembassy.org/
```

```
Created 16 clusters

National (4 documents)
  [ 3] Home: National Portal of India
       http://india.gov.in/

  [12] India : Country Studies - Federal Research Division, Library of ...
       http://rs6.loc.gov/frd/cs/intoc.html

  [20] India Meteorological Department
       http://www.imd.ernet.in/main_new.htm

  [31] India
       http://www.teachers.ash.org.au/jmresources/countries2/india.html

World News (4 documents)
  [14] The Times of India: Latest News India, World & Business News ...
       http://timesofindia.indiatimes.com/

  [15] India News - Breaking World India News - The New York Times
       http://topics.nytimes.com/top/news/international/countriesandterritories/india/index.html

  [21] BBC NEWS | South Asia | Country profiles | Country profile: India
       http://news.bbc.co.uk/2/hi/europe/country_profiles/1154019.stm

  [22] India | World news | guardian.co.uk
       http://www.guardian.co.uk/world/india

Country (3 documents)
  [12] India : Country Studies - Federal Research Division, Library of ...
       http://rs6.loc.gov/frd/cs/intoc.html
```

Figure 4.2.2: Sample clustering output

4.3 The Lingo Clustering Algorithm

Carrot^2 comes with configurable clustering algorithms. The Article Generation Engine uses the Lingo clustering algorithm. The algorithm was developed by Stanisław Osiński, Jerzy Stefanowski, and Dawid Weiss. It operates in following manner:

> "The Lingo Algorithm follows steps of frequent phrase extraction, cluster label induction, cluster content discovery and final cluster formation**.** When designing a web search clustering algorithm, special attention must be paid to ensuring that both content and description (labels) of the resulting groups are meaningful to humans. As stated on Web pages of Vivisimo (http://www.vivisimo.com) search engine, "a good cluster—or document grouping—is one, which possesses a good, readable description". The majority of open text clustering algorithms follows a scheme where cluster content discovery is performed first, and then, based on the content, the labels are determined. But very often intricate measures of similarity among documents do not correspond well with plain human understanding of what a cluster's "glue" element has been. To avoid such problems Lingo reverses this process—we first attempt to ensure that we can create a human-perceivable cluster label and only then assign documents to it. Specifically, we extract frequent phrases from the input documents, hoping they are the most informative source of human-readable topic descriptions. Next, by performing reduction of the original term-document matrix using SVD, we try to discover any existing latent structure of diverse topics in the search result. Finally, we match group descriptions with the extracted topics and assign relevant documents to them.**"**
>
> **-**Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition [17].

5. Summarizer

After the search results have been divided into clusters, the next step in the article generation is to summarize the information in each cluster to present the important information. This section discusses various summarizing approaches.

5.1. OTS

Automatic text summarization is the technique where a computer program summarizes a document. Summarizing of text and collecting important contents from multiple sentences is an important module for the Article Generation Engine.

The Open Text Summarizer [6] is an open source tool for summarizing texts. The program reads a text and decides which sentences are important and which are not. The project uses the OTS version 0.5.0. OTS uses “GNU make” build mechanism. OTS can be run from command line as follows:

```
$ ots articles/sacbee1.txt--html
```

The above command will summarize the sacbee1.txt file and will generate the summarized text output in html format. The highlighted text in Figure 5.1.1 shows the summarized text from the text file.

Output

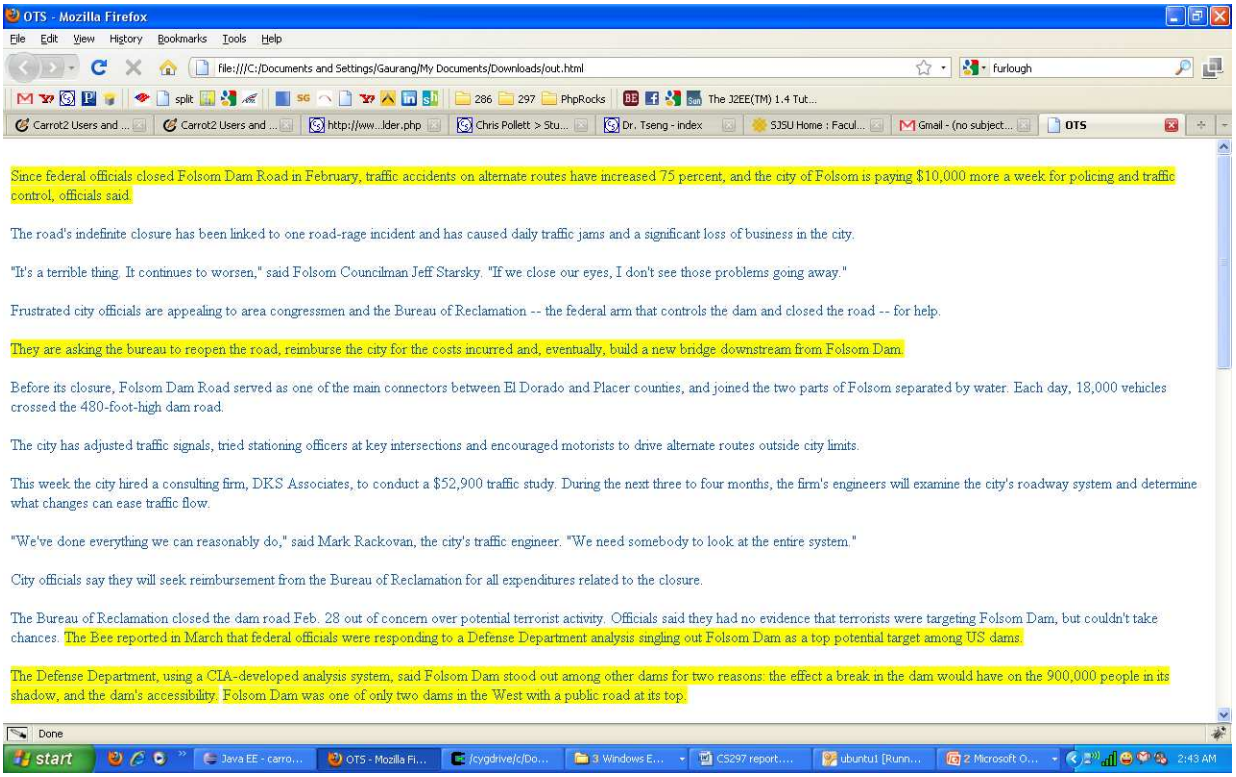

Figure5.1.1: Sample OTS output

5.2 Great Summary

GreatSummary [5] is another summarizing tool, which can summarize web pages. We created our own API on top of GreatSummary as it only has an end-user interface. For an end-user, they can use GreatSummary via these steps:

1. User pastes a text source or URL and identifies the number of sentences to be returned [5].
2. The system identifies the sentences in the text [5].
3. Using a mathematical technique called singular value decomposition; the system identifies the words that capture the key threads of the text. The process is repeated until the number of sentences requested by the user is reached [5].
4. GreatSummary then ranks the sentences according to these words [5].
5. The results are returned to the user [5].

Figure 5.2.1 illustrates summary for web page http://en.wikipedia.org/wiki/India

Online GreatSummary Run

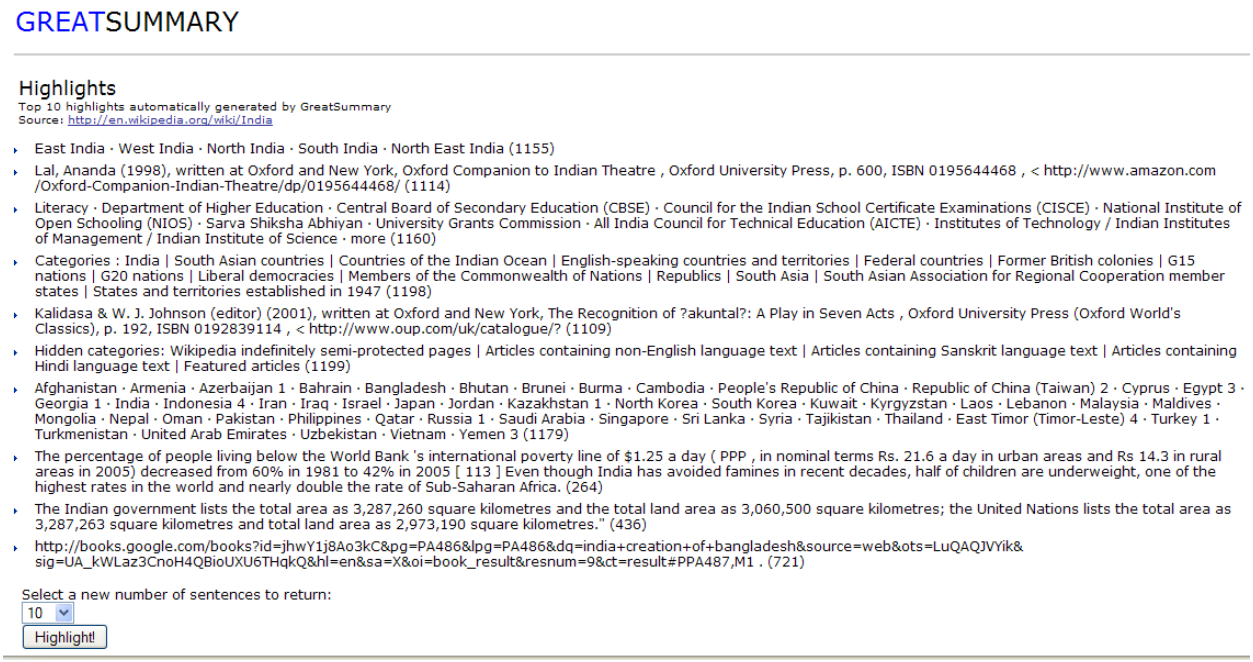

Figure 5.2.1: GreatSummary summarizes web page http://en.wikipedia.org/wiki/India

5.3. Summarizing Using Carrot^2

Here we are looking for the possibility of using Carrot^2 for document level clustering. Document level clustering is basically clustering the contents of a web page to organize the information on that page. The aim of this deliverable is to modify the Carrot^2 code to make it work for document level clustering. The list of documents to be clustered is one of the input parameters to Carrot^2 clustering engine. An API, that breaks a web page into sub documents, was developed. The output of this API can be passed as input to Carrot^2.

Carrot^2 code was explored to find the appropriate place to integrate the new API in the system. While exploring though Carrot^2 codebase, the following observations were made on the flow of the code as illustrated in Figure 5.3.1.

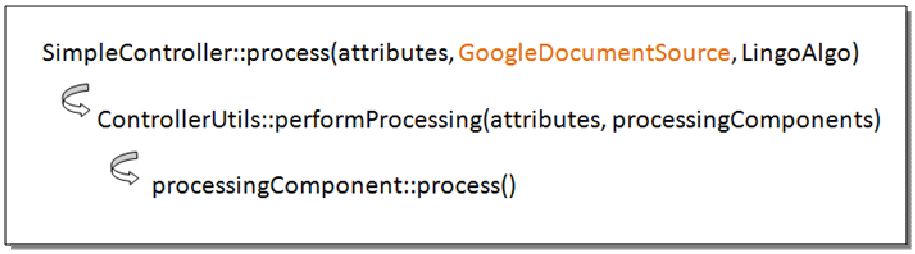

Figure 5.3.1: Flow of clustering code in Carrot^2

The `SampleController`, which is the entry point of clustering, receives GoogleDocumentSource.class as an argument. The GoogleDocumentSource.class is a Java file in a Carrot^2 sub-project named `carrot2-source-google`. This class is responsible for fetching search results from Google and organizing them into list of documents that can be understood by Carrot^2. Carrot^2 has support for several search engines, such as, `carrot2-source-google`, `carrot2-source-microsoft`, `carrot2-source-lucene`, etc. It has separate sub projects for all search engines it supports. Therefore we simply created a new API, named `carrot2-source-document`, which can divide a document into sub documents and generate a list of the documents understandable by Carrot^2.

A new file, ClusteringDocument.java, was created in the `carrot2-examples` project. This example can be run to demonstrate the document content level clustering using the Carrot^2.

A new method `getDocumentsFromFile(String pageURL)` was added to the `ClusteringDocument` class, to divide the inputted page in sub documents and return the list of sub documents understandable by the Carrot^2 clustering algorithm. Figure 5.3.2 show a snippet from the code of this method.

```
        int pageLength = pageString.length();
        int docsCnt = pageLength/500;
        String [][] docContent = new String [docsCnt][3];

        int i =0;
        for(i=0;i<docsCnt;i++) {
                String docText = pageString.substring(i*500, (i+1)*500 -1 < pageLength ? (i+1)*500 -1 :
pageLength-1);
                docContent[i][0] = docText.substring(0,50);
                docContent[i][1] = docText;
                docContent[i][2] = "";
        }

         List<Document> documents = new ArrayList<Document>();
         for (final String [] element : docContent)
             documents.add(new Document(element[0], element[1], element[2]));

        return Collections.unmodifiableList(documents);
```

Figure 5.3.2: Code snippet of `getDocumentFromFile()` method

Clustering output for URL: http://en.wikipedia.org/wiki/Data_mining

Figure 5.3.3 shows the clustering results.

```
Attributes:
processing-time-total:   125
processing-time-algorithm:   125
2009-08-26 15:21:00,937 INFO  org.carrot2.clustering.lingo.LingoClusteringAlgorithm: Native BLAS routines not available
Collected 79 documents


Created 27 clusters

Geographic Data (7 documents)
  [45]  So far, data mining and Geographic Information Sy

  [48] ts, that are conventionally archived in hybrid dat

  [49] de ill-structured data such as imagery and geo-ref

  [50] rability, including differences in semantics, refe

  [51] eraction through attributed geographic space such

  [66] Ã©n GN, Bate A, Hopstadius J, Star K, Edwards IR.

  [67] ., (eds.), 1999, Spatial Multimedia and Virtual Re

International Conference (6 documents)
  [12] Information Technology and Decision Making summari

  [13] her Computer Science conferences on data mining in

  [59]  for the Field of Data Mining and Knowledge Discov
```

Figure 5.3.3: Clustering results for webpage: http://en.wikipedia.org/wiki/Data_mining

6. Automated Article Generation Website

The final product of the CS298 writing project is a website that allows users to enter the query term and see articles. The website was developed using web technologies of PHP, XHTML, CSS, ETS (Easy Template System).

6.1 Website Architecture

Figure 6.1.1 illustrates directory structure of the website. It used backend models for integration of clustering and summarizing system modules. The web site further has various modules like article, summarizer, noise, landing, framework, etc. Each of these modules has their own

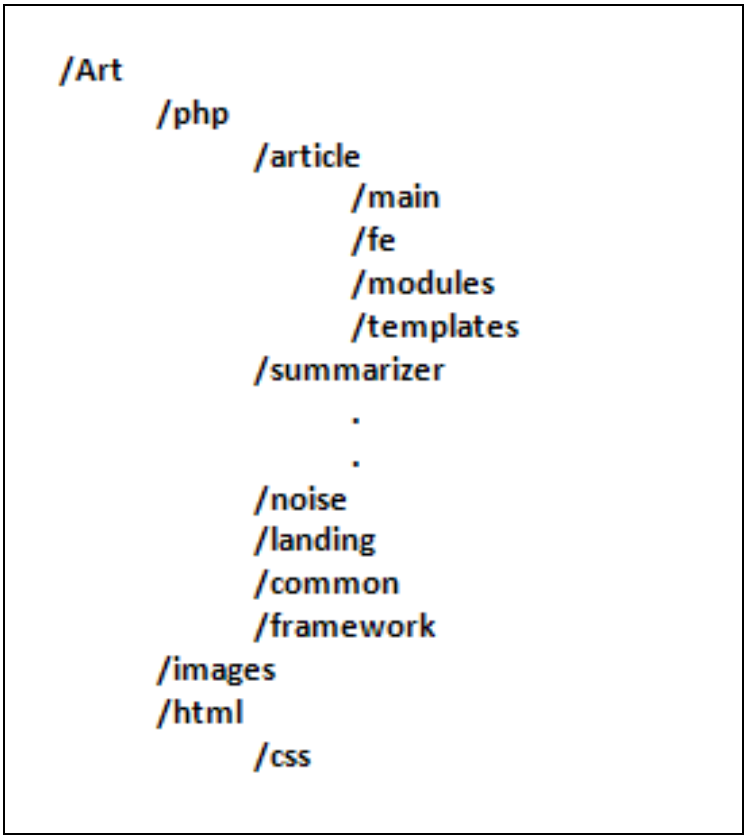

Figure 6.1.1: Directory structure for website

functionalities. The framework module defines the framework that is used throughout the website to render pages. The article module is the main clustering engine.

The website follows MVC architectural pattern for organizing the code. Each module directory has sub directories for MVC components. Directory /main, which is the entry point to the module, is the "controller" in the MVC pattern. Directory /fe and /modules are "view" and "model" MVC components respectively.

ETS- Easy Template System

ETS is a library that allows the creation of HTML templates that are imported and used in PHP scripts [13]. The template files generally reside in the /templates directory.

6.2. Summarizing- A Configurable Model

Different summarizing approaches were experimented to generate better quality articles. The website comes with configurable summarizers. The argument on which summarizer to be used is passed to the constructor of `Summarizer` class in php/summarizer/Summarizer.php. For example, `GreatSummary`, `MixedSummarizer` and `OTS`. The summarizer can also be configured with the function `Summarizer::setSummarizer($name)`, which receives summarizer name as the argument. The MixedSummarizer module combines both the `GreatSummary` and `OTS` modules to produce better summaries. The Figure 6.2.1 illustrates the configurable model of the summarizer.

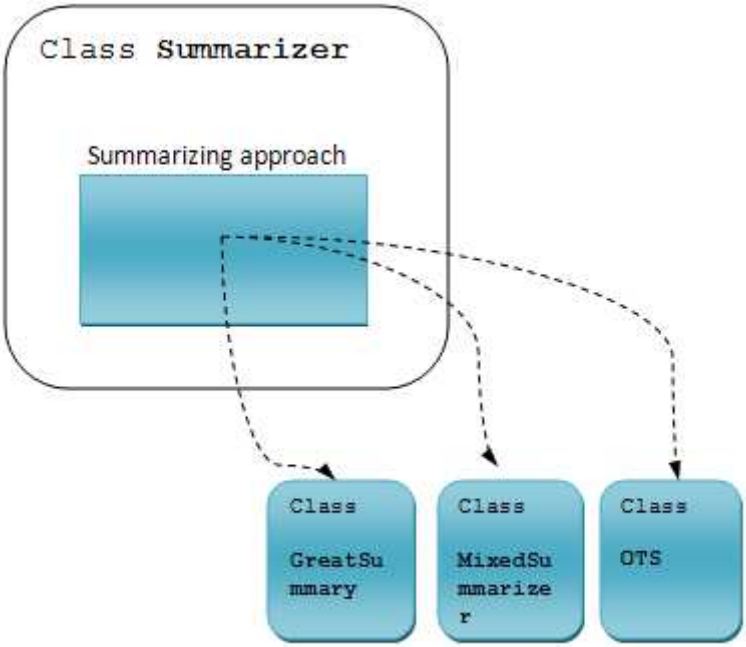

Figure 6.2.1: Configurable summarizers

7. Integrating the Whole System

Integration of different system modules into the website was another milestone of the project. The website is in PHP, while Carrot^2 code is developed in Java and OTS is written in C. Moreover, GreatSummary is an online tool and does not have any APIs. It was a challenging task to integrate all these into one system.

7.1. Integrating Carrot^2 with the Website

The easiest way to use output of a Java program in PHP script is via executing that Java program by command line. PHP has functions, such as, `exec()`, `system()` and `passthru()`, that allow one to execute shell commands. An API method `ArticleAPI::executeCommand()`, which executes a shell command and returns the row output in string format, was developed as shown in Figure 7.1.1.

The function `exec()` of PHP is used in the above function. It executes a shell command and returns the results in the form of an array. The returned array elements are merged into a string using `implode()` function to format the final output.

```
// this functions executes command and returns the output in string format
private static function executeCommand($command) {
    $clusters = array();
    exec($command,$clusters);
    $str = implode("",$clusters);
    return $str;
}
```

Figure 7.1.1: `ArticleAPI::executeCommand()` method

Modified Carrot^2 Output Format

As seen in the Figure 4.2.2, the methods in ClusteringDataFromDocumentSources.java output the clusters in simple text format. We want to have an array of clusters for the PHP code to be able to analyze those clusters further. So we decided to output the Carrot^2 results in a format which was understandable by PHP. The output was formatted in a way that it looks like a PHP array.

The following three methods achieve this:

- `ExampleUtils::displayResultsPHPUnderstandable()`
- `ExampleUtils::displayClusterPHPUnderstandable()`
- `ExampleUtils::displayDocumentPHPUnderstandable()`

Figure 7.1.2 illustrates the code snippets of these methods.

```
public static void displayResultsPHPUnderstandable(ProcessingResult processingResult) {
    final Collection<Cluster> clusters = processingResult.getClusters();

    String clusters1 = "array(\n";
    if (clusters != null) {
        for (final Cluster cluster : clusters) {
            clusters1 += displayClusterPHPUnderstandable(cluster);
            clusters1 += "\n";
        }
        int len = clusters1.length();
        clusters1 = clusters1.substring(0,len-2) + "" + clusters1.substring(len -1);
    }
    clusters1 += ");";
    System.out.print(clusters1);
}

private static String displayClusterPHPUnderstandable(Cluster cluster) {
    final String label = cluster.getLabel();
    String docs = "\tarray(\n\t\t'label' => \"" + label + "\",\n";
    docs += "\t\t'docs' => array(\n";

    for (final Document document : cluster.getDocuments())
    {
        docs += displayDocumentPHPUnderstandable(document);
        docs += "\n";
    }
    int len = docs.length();
    docs = docs.substring(0,len-2) + "" + docs.substring(len -1);

    docs += "\t\t\t\t\t\t)\n";
    docs += "\t),";
    return docs;
}
```

```
private static String displayDocumentPHPUnderstandable(Document document) {
    String doc = "\t\t\t\t\t\tarray(\n";
    doc += "\t\t\t\t\t\t\t'title' => \""  + document.getField(Document.TITLE) + "\",\n";
    doc += "\t\t\t\t\t\t\t'url' => \"" + document.getField(Document.CONTENT_URL) + "\"\n";
    doc += "\t\t\t\t\t\t),";
    return doc;
}
```

Figure 7.1.2: Methods to format Carrot^2 output

Figure 7.1.3 shows one such output string.

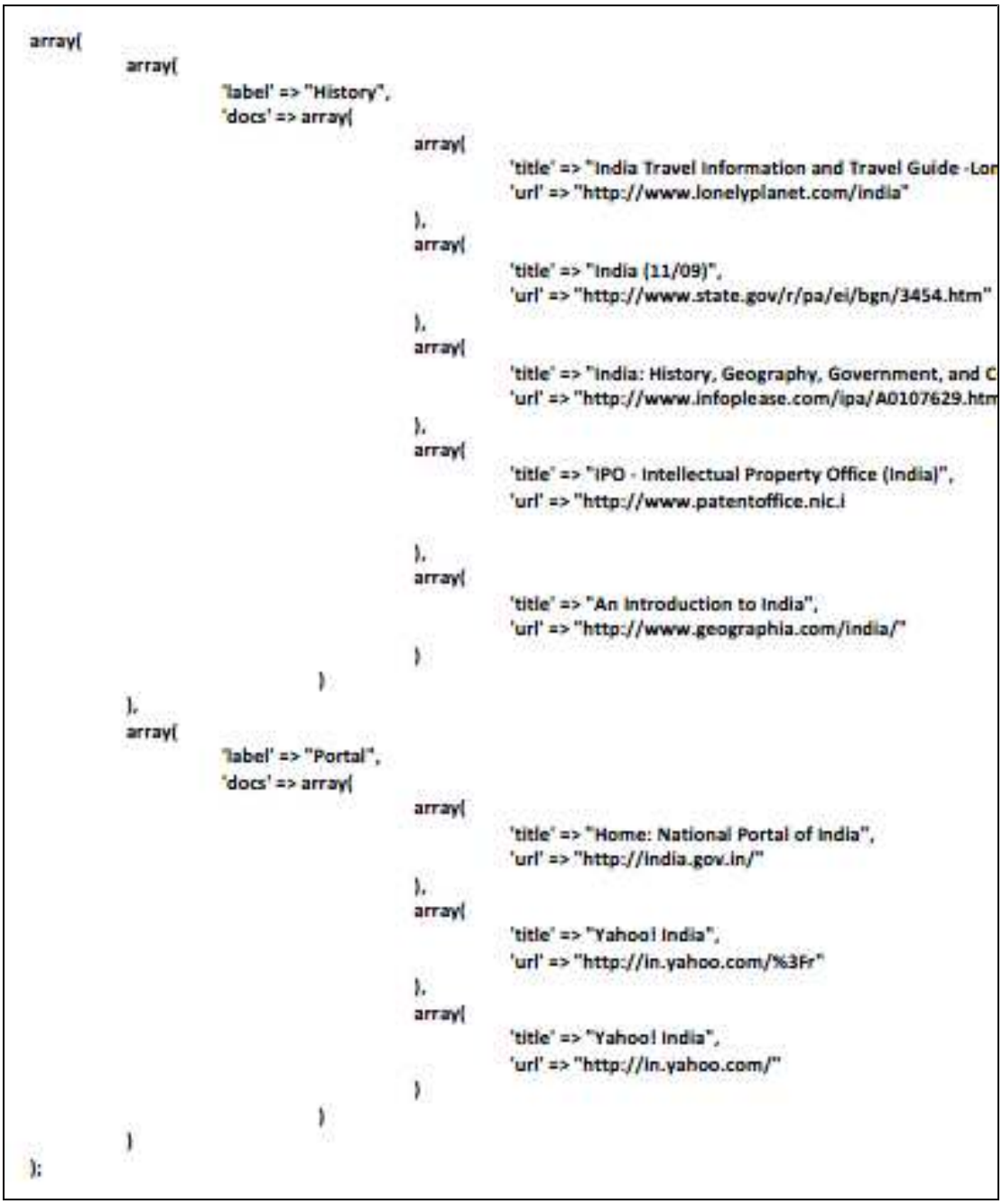

Figure 7.1.3: Output format (string) of Carrot^2 understandable by PHP

The PHP `eval()` function is used to evaluate this string and convert it in PHP array. The variable `$clusters` in Figure 7.1.4 will contain an array of clusters.

```
$str = ArticleAPI::executeCommand($clusteringCommand. " \"$searchTerm\"");
$str = "\$clusters = " .$str;

eval($str);
return $clusters;
```

Figure 7.1.4: Converting Carrot^2 output to PHP array

7.2. Integrating OTS

Open Text Summarizing library is written in C. It can be run from the command line with various options. For example, it might be executed within PHP using `executeCommand()` function as shown in the Figure 7.2.1

```
//OTS command- using shell script
$clusterStr = addslashes($clusterStr);
return ArticleAPI::executeCommand("bash /var/www/Art/temp/echoWeb.sh \"" . $clusterStr . "\" | ots --ratio 5");
```

Figure 7.2.1: OTS integration- command line

This command runs the echoWeb.sh shell script, which outputs the passed in string to Standard Output. The output is then piped to the OTS tool, which summarizes the paragraphs. Figure 7.2.2 shows the echoWeb.sh script.

```
#!/bin/bash
echo $1;
```

Figure 7.2.2: Script echoWeb.sh

After obtaining the summary results, the algorithm generating engine applies noise reduction techniques to remove unimportant text.

7.3. Integrating GreatSummary

GreatSummary is an online tool for summarizing the web pages. The project does not provide API in the current release. Our project implements parsing technique on the GreatSummary online page to obtain the summaries. The cURL library is used in PHP to make request to this web page, which in turn returns the page contents. The web page returned is then parsed to obtain the summary results. Figure 7.3.1 shows the source code of the GreatSummary web page and summarized text being parsed in the rectangle.

The <ul> list pattern is then identified to create a PHP array of summary sentences. These sentences are further processes in the noise reduction module. The resulting article sections are a collection of important sentences from the relevant web pages.

```
                                    <tr>
                                        <td>
                                        <table border="0" width="100%" id="table5">
                                            <tr>
                                                <td>
                                        <ul>

                                        <li>
East India  ·  West India  ·  North India  ·  South India  ·  North East India (1171)
</li>
<li>
Lal, Ananda (1998), written at Oxford and New York,  Oxford Companion to Indian Theatre , Oxford University Press, p. 600,  ISBN 0195644468 , <
http://www.amazon.com/Oxford-Companion-Indian-Theatre/dp/0195644468/ (1130)
</li>
<li>
Literacy  ·  Department of Higher Education  ·  Central Board of Secondary Education (CBSE)  ·  Council for the Indian School Certificate
Examinations (CISCE)  ·  National Institute of Open Schooling (NIOS)  ·  Sarva Shiksha Abhiyan  ·  University Grants Commission  ·  All
India Council for Technical Education (AICTE)  ·  Institutes of Technology  /  Indian Institutes of Management  /  Indian Institute of Science
·  more (1176)
</li>
<li>
Categories :  India  |  South Asian countries  |  Countries of the Indian Ocean  |  English-speaking countries and territories  |  Federal
countries  |  Former British colonies  |  G15 nations  |  G20 nations  |  Liberal democracies  |  Members of the Commonwealth of Nations  |
Republics  |  South Asia  |  South Asian Association for Regional Cooperation member states  |  States and territories established in 1947 (1214)
</li>
<li>
Kalidasa  & W. J. Johnson (editor) (2001), written at Oxford and New York,  The Recognition of ?akuntal?: A Play in Seven Acts , Oxford University
Press (Oxford World's Classics), p. 192,  ISBN 0192839114 , < http://www.oup.com/uk/catalogue/? (1125)
</li>

                                        </ul>
                                                </td>
                                                <td valign="top">
                                        <table width="100%" id="table16">
                                            <tr>
                                                <td>
```

Figure 7.3.1: Source code of GreatSummary web page

8. Noise Reduction

The articles generated from the previous steps often contain useless sentences and text. This noise should be detected and removed to make the article readable and meaningful. The following noise reduction techniques were implemented in our project.

Invisible Text

Invisible text is the code on web pages which is not being displayed on the web page. This includes PHP code, html tags, CSS styles, scripts, applets, embedded frames, etc.

GreatSummary automatically strips invisible text before it applies the summarizing algorithm. Invisible text needs to be removed from the text before passing it to OTS for summarization. Such text should be detected and removed explicitly. We used the function `strip_html_tags()` to remove this kind of text. This is illustrated in Figure 8.1.1.

```
function strip_html_tags( $text )
{
    $text = preg_replace(
        array(
          // Remove invisible content
            '@<head[^>]*?>.*?</head>@siu',
            '@<style[^>]*?>.*?</style>@siu',
            '@<script[^>]*?.*?</script>@siu',
            '@<object[^>]*?.*?</object>@siu',
            '@<embed[^>]*?.*?</embed>@siu',
            '@<applet[^>]*?.*?</applet>@siu',
            '@<noframes[^>]*?.*?</noframes>@siu',
            '@<noscript[^>]*?.*?</noscript>@siu',
            '@<noembed[^>]*?.*?</noembed>@siu',
          // Add line breaks before and after blocks
            '@</?((address)|(blockquote)|(center)|(del))@iu',
            '@</?((div)|(h[1-9])|(ins)|(isindex)|(p)|(pre))@iu',
            '@</?((dir)|(dl)|(dt)|(dd)|(li)|(menu)|(ol)|(ul))@iu',
            '@</?((table)|(th)|(td)|(caption))@iu',
            '@</?((form)|(button)|(fieldset)|(legend)|(input))@iu',
            '@</?((label)|(select)|(optgroup)|(option)|(textarea))@iu',
            '@</?((frameset)|(frame)|(iframe))@iu',
        ),
        array(
            ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ',
            "\n\$0", "\n\$0", "\n\$0", "\n\$0", "\n\$0", "\n\$0",
            "\n\$0", "\n\$0",
        ),
        $text );
    return strip_tags( $text );
}
```

Figure 8.1.1: Function `strip_html_tags()`

Special Characters

UTF-8 encoding is applied to remove the special and junk characters from the web pages. PHP function `utf8_encode()` is used to encode the text.

Footer Links

Many websites have footer links in the form of a list separated by the pipe character (|). The footer conveys no meaning in the article body and should be removed to improve the article quality. We created a regular expression pattern to detect such footers.

RegEx for footer links: `"/(.*\|.*)+/"`

Copyrights text

Many websites have a “Copyrights text” at the bottom of the page. This also conveys no meaning in article body.

RegEx for copyrights: `"/COPYRIGHT.*\d{4}/i"`

Breadcrumbs

Many websites have breadcrumbs, such as, *Home>Electronics>Digital Camera*, at the top of the web page. This kind of text should also be removed from the articles.

RegEx: `"/>/"`

9. Article Generation Run

Figure 9.1 and 9.2 shows an example Article Generation run for the query “San Jose”. The article has various sections each representing different aspect of San Jose.

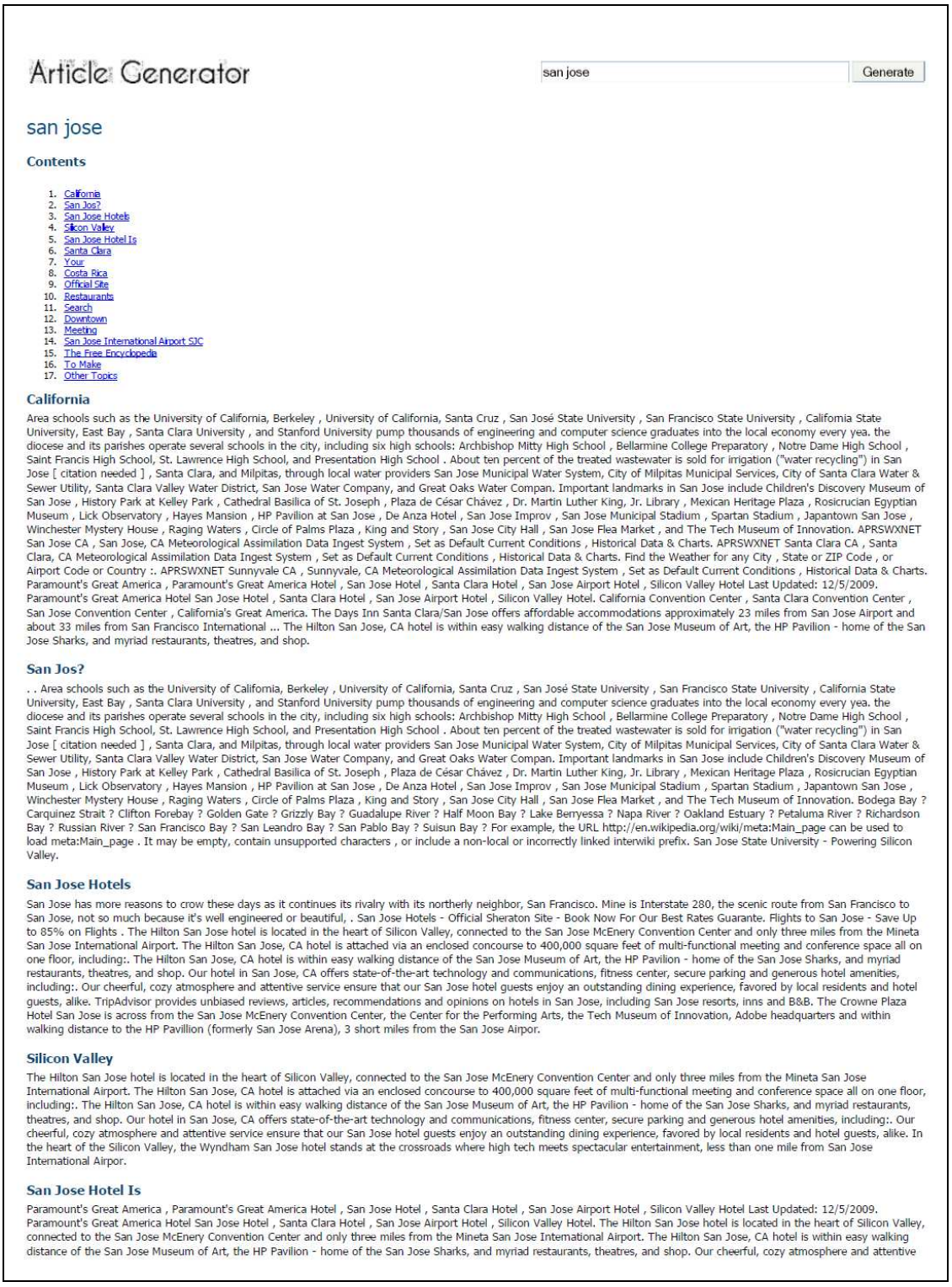

Figure 9.1: Article generation run (paragraph version) for query “san jose” page-1

service ensure that our San Jose hotel guests enjoy an outstanding dining experience, favored by local residents and hotel guests, alike. In the heart of the Silicon Valley, the Wyndham San Jose hotel stands at the crossroads where high tech meets spectacular entertainment, less than one mile from San Jose International Airpor.

Santa Clara

APRSWXNET San Jose CA , San Jose, CA Meteorological Assimilation Data Ingest System , Set as Default Current Conditions , Historical Data & Charts. APRSWXNET Santa Clara CA , Santa Clara, CA Meteorological Assimilation Data Ingest System , Set as Default Current Conditions , Historical Data & Charts. Paramount's Great America , Paramount's Great America Hotel , San Jose Hotel , Santa Clara Hotel , San Jose Airport Hotel , Silicon Valley Hotel Last Updated: 12/5/2009. Paramount's Great America Hotel San Jose Hotel , Santa Clara Hotel , San Jose Airport Hotel , Silicon Valley Hotel. California Convention Center , Santa Clara Convention Center , San Jose Convention Center , California's Great America.

Your

San Jose has more reasons to crow these days as it continues its rivalry with its northerly neighbor, San Francisco. Mine is Interstate 280, the scenic route from San Francisco to San Jose, not so much because it's well engineered or beautiful, . CLARION HOTEL SJ AIRPORT CROWNE PLAZA SAN JOSE DOLCE HAYES MANSION DOUBLETREE HOTEL SAN JOSE FAIRMONT SAN JOSE HILTON SAN JOSE & TOWERS HOLIDAY INN SAN JOSE HOTEL DE ANZA HOTEL MONTGOMERY RADISSON PLAZA HOTEL SAINTE CLAIRE, LARKSPUR SAN JOSE MARRIOTT WYNDHAM HOTEL. CALIFORNIA STATE ASSOC THEATRE & ENTERTAINMENT SMALL/MEDIUM SIZED PROGRAMS NAT'L ASSOC NAT'L ASSOC CA NAT'L ASSOC DC AREA NAT'L ASSOC MD/VA/SW NAT'L CORP WEST NAT'L CORP E.

Costa Rica

For example, the URL http://en.wikipedia.org/wiki/meatball:WikiPedia will give this error, because the "meatball:" interwiki prefix is not marked as local in the interwiki table. However, non-local interwiki pages can still be accessed by interwiki linking or by entering them in the search box. an attempt to load a URL pointing to a "non-local" interwiki page (usually those not run by the Wikimedia Foundation). You may be able to locate the desired page by searching for its name (with interwiki prefix, if any) in the search bo. It may be empty, contain unsupported characters , or include a non-local or incorrectly linked interwiki prefix. Certain interwiki prefixes are marked as local in the table. This joint initiative of the U.S. Department of State and the U.S. Department of Education is part of the U.S. government effort to promote programs in the United States and overseas that prepare Americans for a global environment and that attract future leaders from abroad to experience the United States as they study learn, and exchange experience.

Official Site

As part of the City of San Jose's Budget Reduction Plan four of the City of San Jose's regional parks, Alum Rock Park, Almaden Lake Park, Overfelt Gardens, and Prusch Farm Park, will close on Mondays beginning November 30, 2009. The Guadalupe River Park, Kelley Park and Lake Cunningham Park are not affected by this action. The City of San José's Trail Count 2009 shows city trail usage up by 9.6% in both bike and pedestrian traffic with the highest increase found on the Guadalupe River Trail at Coleman Avenue. On park closure days, visitors, joggers and picnickers will be unable to enter the selected sites and Los Alamitos trail users will be advised of an alternative route. San Jose Sharks and sanjosesharks.com are trademarks of The San Jose Sharks hockey club.

Restaurants

Book A Flight Book A Hotel Rent A Car Book A Cruise Book A Package Book An Activity. San Jose has more reasons to crow these days as it continues its rivalry with its northerly neighbor, San Francisco. Mine is Interstate 280, the scenic route from San Francisco to San Jose, not so much because it's well engineered or beautiful, . San Jose Hotels Get Our Best Price Guarantee on All Hotels in San Jose at Expedi.

Search

Marketers, corporate decision makers, webmasters and search engine marketers (SEMs), including pay-per-click (PPC) advertisers and search engine optimization (SEO) professionals have been attending SES in California for the past 12 years!. Organized and programmed by the SES Advisory Board and SearchEngineWatch.com , the conference will be packed with 70+ sessions covering PPC management, keyword research, SEO, social media, local, mobile, link building, duplicate content, multiple site issues, video optimization and usability, plus an expo floor with 100+ companies to help you grow your business, networking events and mor. Within San Jose itself, you'll find a large concentration of fairly affordable apartments near San Jose State University. Unless you've been living in a nice, affordable cave, it will come as no surprise to you that San Jose and the South Bay have the highest cost of living in the nation, approximately 360% of the national average. Where the Jobs Are As the capital of Silicon Valley, San Jose itself has more than 350,000 jobs and 25 companies with 1,000 or more employees. While viewing your Maps or Directions on MapQuest.com, you can now easily display Hotels, Parking Garages, Restaurants, Gas Stations and more with one simple click using the new On-Map Search Tool!. Easily Access International Directions & Maps On MapQuest Easily Find Events, Local Reviews, Movies, and More!.

Downtown

Great holiday gift items now available!. Give the gift of Downtown Ice to children through our KidSkate progra.

Meeting

CLARION HOTEL SJ AIRPORT CROWNE PLAZA SAN JOSE DOLCE HAYES MANSION DOUBLETREE HOTEL SAN JOSE FAIRMONT SAN JOSE HILTON SAN JOSE & TOWERS HOLIDAY INN SAN JOSE HOTEL DE ANZA HOTEL MONTGOMERY RADISSON PLAZA HOTEL SAINTE CLAIRE, LARKSPUR SAN JOSE MARRIOTT WYNDHAM HOTEL. CALIFORNIA STATE ASSOC THEATRE & ENTERTAINMENT SMALL/MEDIUM SIZED PROGRAMS NAT'L ASSOC NAT'L ASSOC CA NAT'L ASSOC DC AREA NAT'L ASSOC MD/VA/SW NAT'L CORP WEST NAT'L CORP E. Free 'Green' Parking at The Fairmont San Jose: Overnight guests who drive hybrid vehicles can enjoy free parking at The Fairmont San Jos.

San Jose International Airport SJC

In the heart of the Silicon Valley, the Wyndham San Jose hotel stands at the crossroads where high tech meets spectacular entertainment, less than one mile from San Jose International Airpor. ByRequest benefits exclusively at Wyndham Hotels and Resort.

The Free Encyclopedia

Area schools such as the University of California, Berkeley , University of California, Santa Cruz , San José State University , San Francisco State University , California State University, East Bay , Santa Clara University , and Stanford University pump thousands of engineering and computer science graduates into the local economy every yea. Important landmarks in San Jose include Children's Discovery Museum of San Jose , History Park at Kelley Park , Cathedral Basilica of St. Joseph , Plaza de César Chávez , Dr. Martin Luther King, Jr. Library , Mexican Heritage Plaza , Rosicrucian Egyptian Museum , Lick Observatory , Hayes Mansion , HP Pavilion at San Jose , De Anza Hotel , San Jose Improv , San Jose Municipal Stadium , Spartan Stadium , Japantown San Jose , Winchester Mystery House , Raging Waters , Circle of Palms Plaza , King and Story , San Jose City Hall , San Jose Flea Market , and The Tech Museum of Innovation.

To Make

CLARION HOTEL SJ AIRPORT CROWNE PLAZA SAN JOSE DOLCE HAYES MANSION DOUBLETREE HOTEL SAN JOSE FAIRMONT SAN JOSE HILTON SAN JOSE & TOWERS HOLIDAY INN SAN JOSE HOTEL DE ANZA HOTEL MONTGOMERY RADISSON PLAZA HOTEL SAINTE CLAIRE, LARKSPUR SAN JOSE MARRIOTT WYNDHAM HOTEL. CALIFORNIA STATE ASSOC THEATRE & ENTERTAINMENT SMALL/MEDIUM SIZED PROGRAMS NAT'L ASSOC NAT'L ASSOC CA NAT'L ASSOC DC AREA NAT'L ASSOC MD/VA/SW NAT'L CORP WEST NAT'L CORP E. The San Jose Marriott Hotel is alive with the vitality that comes from being located in the heart of San Jose's Business.

Other Topics

Homegrown excellence: Two of Symphony Silicon Valley's best players shine as soloists in the orchestra's weekend program. Authorities are investigating a violent home invasion in Saratoga that injured a 96-year-old father and his 64-year-old daughter late Thursday night. SEC investigation of New York hedge fund Galleon also ensnares the fund manager's network of Silicon Valley friends, associate.

Figure 9.2: Article Generation run (paragraph version) for query “san jose” page-2

10. Results and Limitations

Our text-only article generation can be thought as a first step towards creating a complete and accurate Article Generation Engine. The following are a few interesting comparisons between the Automated Article Generation Engine and some other knowledge engines.

- Wikipedia
    - As Wikipedia articles are generated by users, it is likely to omit articles on some specific topics. For example, “luna moped” was a widely used moped in India during 1990s. Google returns 0.1 million results for the query “luna moped”. In spite of this term being so popular, Wikipedia does not have an article for it. Nevertheless, our Article Generation Engine can generate article on “luna moped”. The Automated Article Generated System can thus generate articles on very specific topics such as, geographically local things, person names, etc.
    - Sometimes people who have a strong opinion about a subject will try to control the articles about that subject. Thus articles on Wikipedia or similar websites might be biased. This problem is potentially reduced with the Automated Article Generator as it receives most relevant search results from Google or Nutch.
- Wolfram|Alpha (http://www.wolframalpha.com)
    - Wolfram|Alpha is a computational knowledge engine. It generates output by doing computations from its own internal knowledge base, instead of searching the web and returning links [16]. It tends to generate visual results rather than text based results. On the other hand, the Article Generation Engine focuses on generating text based articles.

- Automation

  Imagine we want to know everything about “Michael Jackson”. The following are the steps of one of the possible approaches for solving this problem without our system:

  1. Search for “Michael Jackson” on www.google.com.
  2. Explore some of the top results to know about him.
  3. The knowledge gained while exploring the results will leave an impression of who is Michael Jackson in one’s mind.

  Our Article Generation Engine automates the above steps. It is an effort to directly present the user with the impression mentioned in step 3.

10.1 Comparison Statistics

We next compare our articles with the static articles of Wikipedia. The comparison is based on three parameters: (1) The number of schematically similar sections, (2) Interesting information found in our article that Wikipedia does not have and (3) Interesting information found in Wikipedia that our article does not have. To perform the tests we observed articles generated by both Article Generator and Wikipedia for five input queries, which are either names of programming languages or computer science terms. The terms used were Java programming language, Prolog, RDBMS, Scala programming language and C++.

10.1.1 Section Similarity

Here we are observing the number of sections in our article semantically matching with sections in Wikipedia articles. The section names might not match exactly, but they should convey analogous meanings. The following are the sections similarity statistics for each query term.

1) Java Programming Language

Table 10.1.1 shows similar sections found in the two articles for query "Java Programming Language".

Section from Automated generated Article	Similar section in Wikipedia article
1. Tutorial	Examples
2. Resources	See also, References
3. Third edition 4. Fourth Edition	Editions
5. Guide Java	Documentation

Table 10.1.1.1: Similar sections in the articles for the query "Java programming language"

Figure 10.1.1.1 represents the section similarity in form of a simple venn diagram. Two circles in the figure shows the sets of sections in respective articles. Here, AAG refers to Automated Article Generation.

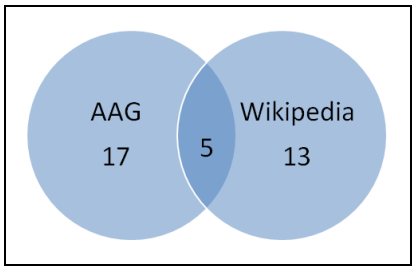

Figure 10.1.1.1: Venn diagram for the query "Java Programming language"

The venn diagram indicates that AAG generated article has 17 sections while Wikipedia article has 13 sections in total. Five sections from the two articles overlap, which is about 33% of the total sections. It means 33% of the sections from the two articles are semantically similar.

2) Prolog

Section from Automated generated Article	**Similar section in Wikipedia article**
1. Prolog tutorial	Examples

Table 10.1.1.2: Similar sections in the articles for the query "Prolog"

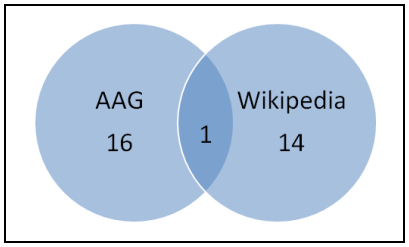

Figure 10.1.1.2: Venn diagram for the query "Prolog"

Here 7% of the total article sections are semantically similar.

3) RDBMS

Section from Automated generated Article	Similar section in Wikipedia article
1. SQL	Structured Query Language(SQL)
2. What is RDBMS 3. A Relational Database Management System	Introduction

Table 10.1.1.3: Similar sections in the articles for the query "RDBMS"

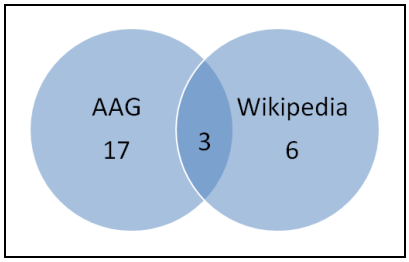

Figure 10.1.1.3: Venn diagram for the query "RDBMS"

Here 26% of the total article sections overlap.

4) Scala Programming Language

Section from Automated generated Article	Similar section in Wikipedia article
1. Object oriented and functional	Object-oriented features, Functional programming

Table 10.1.1.4: Similar sections in the articles for the query "Scala Programming Language"

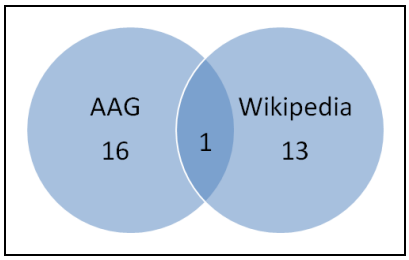

Figure 10.1.1.4: Venn diagram for the query "Scala Programming Language"

Here 7% of the total article sections are semantically similar.

5) C++

Section from Automated generated Article	**Similar section in Wikipedia article**
1. C Libraries 2. Library	Standard library
3. C compilers	List of C++ Compilers
4. The programming language	Introduction

Table 10.1.1.5: Similar sections in the articles for the query "C++"

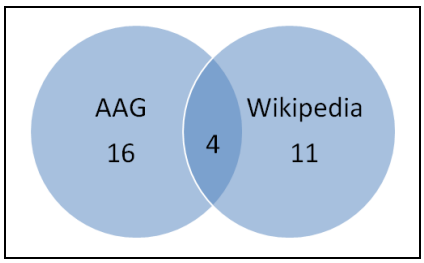

Figure 10.1.1.5: Venn diagram for the query "C++"

Here 30% of the total article sections are semantically similar.

Averaging the percentage for the above observations results in the value of 21%, which means 21% of the sections in our articles are semantically similar with the sections in the Wikipedia articles.

10.1.2 Text Similarity

Semantic similarity analysis between sentences in both articles can be done to compare the text similarity between the articles. The analysis includes techniques of LSA (Latent semantic analysis), Terminology extraction, PMI (Point wise Mutual Information), etc. Such a detailed analysis is beyond the scope of this project.

Here we are conducting a simple comparison of content/sentences between the two articles. The aim is to observe mutually exclusive information from both the articles. Following are some of the observations for the sample queries.

1) Query: Java Programming Language

- The following information is present in our article, but not in the Wikipedia article:
  - Information on various Java books is found in the "Books" section of the article, while no information on books is present in the Wikipedia article.
  - "With the Java Media Framework API, Java now has excellent multimedia playback and encoding capabilities." The sentence talks about media specific features of Java language.

- "Performance" section in the Wikipedia article discusses JVM execution speed and performance issues. The AAG article does not have sentences talking on performance.
- Moreover, the AAG article has sentences talking about the Memory model. Similar information is conveyed in the "Automatic memory management" section in the Wikipedia article.

2) *Query: C++*

- The following information is present in our article, but not in the Wikipedia article:
  - "Dynamic memory allocation : blocks of memory of arbitrary size can be requested at run-time using library functions such as malloc from a region of memory called the heap ; these blocks persist until subsequently freed for reuse by calling the library function free." It talks about `malloc` function of C++ library.
  - "Initialization lists are necessary for most classes that use inheritance or include objects." The sentence talks about initializing an object while inheriting.
  - "C++ History - http://www.hitmill.com/programming/cpp/cppHistory.html" This URL has useful information on the C++ history.
  - "In 1983, the American National Standards Institute (ANSI) formed a committee, X3J11, to establish a standard specification of C. In 1989, the standard was ratified as ANSI X3.159-1989 "Programming Language C." This version of the language is often referred to as ANSI C, Standard C, or sometimes C8." The statement states the evolvement of ANSI standard of C language.

- Criticisms of the C++ language are discussed in a separate section in the Wikipedia article, but the AAG article does not mention such criticisms.

- The C++ standard and other libraries are mentioned in the “Standard Library” section of the Wikipedia article. The AAG article also narrates different libraries at several places in the article.

Similar comparisons may be observed for other queries. Observations show that most of the information in both the articles is semantically same. However, both of the articles have certain information which is not found in their counterparts.

The dynamically generated articles are superior in the sense that they include more specific details compared to the Wikipedia articles. For example, the dynamic memory allocation issue raised in the article on C++ is not found in the corresponding Wikipedia article. Another example is the information on Java books and the Media framework API found in the article on Java programming language. On the other hand, these details are sometimes listed with some unrelated sentences in the AAG article. For example, the sentence talking about the ANSI standard in the C++ article would have been more meaningful if the article was on the C Programming language. Furthermore, the information in the Wikipedia articles has a better flow than AAG articles.

Moreover, the Article Generation Engine produces duplicate contents. Based on the careful observation of the articles, it can be concluded that 80% of the text in the article is unique. The remaining 20% of the text is the repetition of sentences.

10.2 Limitations of AAG Generated Article

- The generated articles are comprised of text data only. The Article Generation Engine does not consider images and other form of data.
- Although the AAG articles convey the gist of the relevant section, the section content does not flow to the degree that can match with the level of hand written content/paragraphs.

11. Conclusion

Automated Article Generation from information available on the web is a new direction as to how the articles are generated currently. No authors, no writing, no editing is needed. Our Article Generation Engine can generate articles on very specific topics, which are likely to be omitted by static articles like Wikipedia. Moreover the articles include some tiny details on the topic which are not found in the static articles. The articles are not as organized and continuous as static articles though. Nevertheless, the engine is able to mine relevant information into well defined sections with the similarity of 21% with the sections in the Wikipedia articles. Further improvements to the engine may enable it to generate competitive articles to those on famous websites. There are performance issues with the engine. Generation is relatively slow because of the time cURL takes to fetch the documents. Efficient caching strategies can be implemented around cURL to avoid repeated fetches of pages.

12. References

[1] Carrot^2 Clustering Engine. Feb. 6, 2006. [Online]. Available: www.carrot2.org/. [Accessed: Jan-Dec, 2009].

[2] Carrot^2 API documentation. Feb. 6, 2006. [Online]. Available: http://download.carrot2.org/stable/javadoc/ [Accessed: Jan-Dec, 2009].

[3] Carrot^2 Users and Developers forum & mailing list archive. Feb. 6, 2006. [Online]. Available: http://project.carrot2.org/forum.html [Accessed: Jan-Dec, 2009].

[4] Carrot^2 at Sourceforge.net. July 17, 2003. [Online] Available: http://sourceforge.net/projects/carrot2/ [Accessed: Jan-Dec, 2009].

[5] GreatSummary - Just the Highlights. March 24, 2007. Available: http://www.greatsummary.com/ [Accessed: Jan-Dec, 2009].

[6] Open Text Summarizer. Aug. 1, 2003. Available: http://libots.sourceforge.net/ [Accessed: Jan-Dec, 2009].

[7] Nutch Wiki. May 29, 2006. Available: http://wiki.apache.org/nutch/NutchTutorial [Accessed: Jan-Dec, 2009].

[8] Nutch Crawler. Jun. 4, 2005. Available: http://lucene.apache.org/nutch/ [Accessed: Jan-Dec, 2009].

[9] Jana Kocibova, Karel Klos, Ondrej Lehecka, Milos Kudelka, and Vaclav Snasel. "Web Page Analysis: Experiments Based on Discussion and Purchase Web Patterns," IEEE/WIC/ACM International Conferences on Web Intelligence, 2007.

[10] Hao Han, and Takehiro Tokuda. "A Method for Integration of Web Applications Based on Information Extraction," Eighth International Conference on Web Engineering, 2007.

[11] Gang Zhang, Yue Liu, Songbo Tan, and Xueqi Cheng."A Novel Method for Hierarchical Clustering of Search Results," IEEE/WIC/ACM International Conferences on Web Intelligence, 2007.

[12] PHP: Hypertext Preprocessor. July 1, 1998. Available: http://php.net/ [Accessed: Aug-Nov, 2009].

[13] ETS- Easy Template System. Sep. 22, 2002. Available: http://ets.sourceforge.net/ [Accessed: Aug-Nov, 2009].

[14] Noise Reduction- Remove invisible text. Oct. 11, 2007. Available: http://nadeausoftware.com/articles/2007/09/php_tip_how_strip_html_tags_web_page [Accessed: Sept, 2009].
[15] PHP: cURL Manual. Apr. 13, 2008. Available: http://php.net/manual/en/book.curl.php [Accessed: Sept, 2009].

[16] Wolfram|Alpha. Available: http://www.wolframalpha.com [Accessed: Jan-Dec, 2009].

[17] Stanisław Osiński, Jerzy Stefanowski, and Dawid Weiss. "Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition," Institute of Computing Science, Pozna´n University of Technology, Poland. 2004.

[18] DMOZ Open Directory. Jan. 25, 1999. Available: http://www.dmoz.org [Accessed: March, 2009].