MSCI 541 – HW2

Due: Noon, Friday, Feb 17, 2012

Date: Monday, Feb 26, 2012 (used one late day)

Name: Joonha Shin

ID: 20236506

## Results & Discussion & Implementation Issues

I regret to say I was not able to finish this assignment. I am under the impression that it was because I still lack enough programming experience required to complete this assignment. I don’t think it was because I didn’t understand enough of the course material, because I knew that I wanted to do, but I couldn’t do it because I ran into issues regarding memory management, concerns about writing clean code and doing things the “right” way. Simply put, I think I was too concerned with doing things right, rather than focusing on doing the right thing.

I ended up circling back and forth between adding few new lines of code, and then changing codes I’ve already written over and over, under the pretense that I was ‘improving’ it, when in fact, I may have been just wasting time. As I did this, I realized I wasn’t making progress, which discouraged me and caused me to lose motivation, causing my performance to greatly suffer.

I believe I spent a fair amount of time on this, although it can be argued that I should have tried harder. I was disappointed for not being able to complete this assignment and realized that I need to do things differently for the next one. I know that other members of this class were able to finish this assignment even before reading week, so the expectation is that I should be able to as well. I have failed to meet this expectation, and for that, I am sorry.

I am under the impression that further assignments will get even tougher. I have tried my best working on this alone and realized where I stand in terms of my limits when it comes to programming. From this point forward, I am going to seek help from my fellow classmates, teaching assistant, or from the professor, not just on programming design choices, but also on fixing negative programming habits.

The source code of my work so far can be found at [www.github.com/jayfiveshin/msci\_541\_hw\_2](http://www.github.com/jayfiveshin/msci_541_hw_2).

## Methods

My plan was to read-in latimes.dat.gz one line at a time and store data from each line to a “document” variable. Once it reaches the end of the document (</DOC> tag), it should begin tokenizing the document. From here, while tokenizing, the program should grab information about the document length, and start building the hash table for the index. I planned to do this for every document in the collection, until the index was complete.

I was not able to satisfy task 1. I experimented enough to write codes that will satisfy task 2 and I could not solve a minor bug associated with satisfying task 3. I did not have a chance to satisfy task 4 and 5, although I have an idea of how I would try to do it.

## Source Code

s**earch\_engine.rb**

require 'zlib'

require 'stemmify'

class SearchEngine

def read\_gzip(file\_name)

Zlib::GzipReader.open(file\_name).read

end

def write\_to\_file

hash = {"la10189-0001" => {"joonha" => 1, "shin" => 1}, "la10189-0002" => {"jake" => 2, "nolan" => 1}}

open("testing.txt", "w") { |f|

hash.each { |k, v|

f.write "<DOC>\n"

f.write "<DOCNO>#{k}</DOCNO>\n"

f.write "<INDEX>\n"

v.each { |k, v|

f.write "#{k}=#{v}\n"

}

f.write "</INDEX>\n"

f.write "</DOC>\n"

}

}

end

def read\_from\_file

mid\_of\_doc = false

hash = Hash.new

docno = String.new

open("testing.txt").each do |line|

line.downcase!

if line.match("<doc>")

mid\_of\_doc = true

elsif mid\_of\_doc and line.match("<docno>")

docno = line.tokenize.join

hash[docno] = {}

elsif mid\_of\_doc and line.match("<index>")

next

elsif mid\_of\_doc and line.match("</doc>")

mid\_of\_doc = false

elsif mid\_of\_doc

terms = line.tokenize

hash[docno][terms[0]] = terms[1]

end

end

puts hash

end

def invert(str)

str.tokenize.each { |word|

@hash[word] = @hash[word].to\_i + 1

}

@hash

end

def get\_docno(str)

docno = String.new

str.downcase!

if str.match "<docno>"

docno = str.tokenize[0]

# docno = "#{docno[0]} #{docno[1]}"

# print "\r\e[0K#{docno}"

@hash[docno] = {}

end

@hash

end

end

class String

def tokenize

term = String.new

terms = Array.new

mid\_of\_tag = false

self.downcase.split("").each { |char|

if char === '>'

mid\_of\_tag = false

elsif mid\_of\_tag

next

elsif char === '<'

mid\_of\_tag = true

elsif char.match(/[a-z0-9]/)

term += char

elsif term.empty?

next

else

terms << term.stem

term.clear

end

}

if term.empty?

terms

else

terms << term.stem

end

end

end