Lucie MICHELET

Telecommunications Software Final Exam

Exercise 1 : Suppose for TCP's RTO estimation, that EstimatedRTT is 4.0 at some point and subsequent measured RTTs all are 1.0. If the initial value of Deviation was 0.5, how long does it take before the TimeOut value, as calculated by the Jacobson/Karels algorithm, falls below 4.0? Use =1/8, Ø = 4. (Hints: use the following equations and loop calculation to minimize TimeOut.)

$$
\begin{aligned}
Difference &= SampleRTT - EstimatedRTT \\
EstimatedRTT &= \alpha \times EstimatedRTT + (1-\alpha) \times SampleRTT \\
Deviation &= Deviation + \delta(|Difference| - Deviation) \\
TimeOut &= \mu \times EstimatedRTT + \phi \times Deviation
\end{aligned}
$$

With α = 0.9 it takes 23 iterations before time out goes below 4.

With α = 0.875 it takes 19 iterations before time out goes below 4.

```
alpha = 7/8
sigma = 1/8
teta = 4
phi = 1

sample = 1
est = 4
dev = 0.75
Time = phi*est + teta*dev
i =0

while Time > 4 :
    dif = sample - est
    est = alpha*est + (1-alpha)*sample
    dev = dev + sigma*(abs(dif)-dev)
    Time = phi*est + teta*dev
    i +=1
```

Name	Type	Size	Value
alpha	float	1	0.9
dev	float	1	0.657156027006192
dif	float	1	-0.29543127065508434
est	float	1	1.2658881435895761
i	int	1	23
phi	int	1	1
sample	int	1	1
sigma	float	1	0.125
teta	int	1	4
Time	float	1	3.8945122516143442

Exercise 2 :

```
GET /cs453/index.html HTTP/1.1<cr><lf>Host: gai
a.cs.umass.edu<cr><lf>User-Agent: Mozilla/5.0 (
Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gec
ko/20040804 Netscape/7.2 (ax) <cr><lf>Accept:ex
```

```
t/xml, application/xml, application/xhtml+xml, text
/html;q=0.9, text/plain;q=0.8, image/png,*/*;q=0.5
<cr><lf>Accept-Language: en-us, en;q=0.5<cr><lf>Accept-
Encoding: zip, deflate<cr><lf>Accept-Charset: ISO
-8859-1, utf-8;q=0.7,*;q=0.7<cr><lf>Keep-Alive: 300<cr>
<lf>Connection:keep-alive<cr><lf><cr><lf>
```

1a. What is the URL of the document requested by the browser?

/cs453/inex.html

```
GET /cs453/index.html
```

1b. What version of HTTP is the browser running?

1.1 `HTTP/1.1`

1c. Does the browser request a non-persistent or a persistent connection? Keep-alive so **Persistent**

1d. What is the IP address of the host on which the browser is running?
Gaia.cs.umass.edu(128.119.245.12)

1e. What type of browser initiates this message? Why is the browser type needed in an HTTP request message?

Mozilla/5.0, the type of browser that initiate the message is **HTML.**

```
HTTP/1.1 200 OK<cr><lf>Date: Tue, 07 Mar 2008
12:39:45GMT<cr><lf>Server: Apache/2.0.52 (Fedora)
<cr><lf>Last-Modified: Sat, 10 Dec2005 18:27:46
GMT<cr><lf>ETag: "526c3-f22-a88a4c80"<cr><lf>Accept-
Ranges: bytes<cr><lf>Content-Length: 3874<cr><lf>
Keep-Alive: timeout=max=100<cr><lf>Connection:
Keep-Alive<cr><lf>Content-Type: text/html; charset=
ISO-8859-1<cr><lf><cr><lf><!doctype html public "-
//w3c//dtd html 4.0 transitional//en"><lf><html><lf>
<head><lf> <meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1"><lf> <meta
name="GENERATOR" content="Mozilla/4.79 [en] (Windows NT
5.0; U) Netscape]"><lf> <title>CMPSCI 453 / 591 /
NTU-ST550ASpring 2005 homepage</title><lf></head><lf>
<much more document text following here (not shown)>
```

2a. Was the server able to successfully find the document or not? What time was the document reply provided?

`200 OK<` is the response of the server. The document was find **successfully.**

Time : Tue, 07 March 2008, 12:39:45

2b. When was the document last modified?

`>Last-Modified: Sat, 10 Dec2005 18:27:46` Saturday 10th December 2005 at 18:27:46

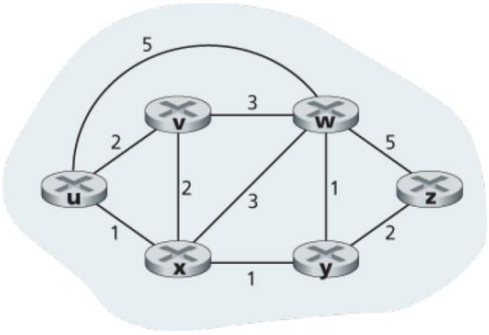
2c. How many bytes are there in the document being returned?

`·Content-Length: 3874‹` **3874 bytes.**

2d. What are the first 5 bytes of the document being returned? Did the server agree to a persistent connection?

The first bytes : <!doc, the server agrees to persistent connection because keep-alive.

Exercise 3 :



1. Looking at Figure 5.3, enumerate the paths from y to u that do not contain any loops.

All the paths are : YXU // YWXU // YWXVU // YZWVU // YZWXVU

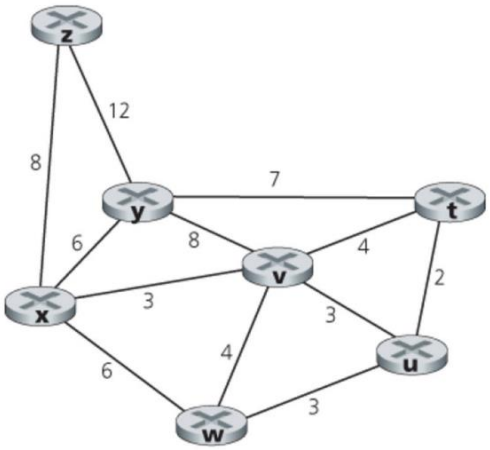
2. Repeat Problem 1 for paths from x to z, z to u, and z to w.

X to Z : XYZ // XWZ // XYZ // XVWZ // XWYZ // XUVWZ // XUVWYZ

Z to U : ZYXU // ZWVU // ZWXVU // ZYXVU // ZYWXVU // ZYXWVU

Z to W : ZW // ZYW // ZYXW // ZYXVW // ZYXUVW

3. Consider the following network. With the indicated link costs, use Dijkstra's shortest-path algorithm to compute the shortest path from x to all network nodes. Show how the algorithm works by computing a table similar to Table 5.1.




```
Distance from  x  to  u is 6
Distance from  x  to  v is 3
Distance from  x  to  w is 6
Distance from  x  to  x is 0
Distance from  x  to  y is 6
Distance from  x  to  z is 8
Distance from  x  to  t is 7
```

4. Consider the network shown in Problem P3. Using Dijkstra's algorithm, and showing your work using a table similar to Table 5.1 , do the following:

a. Compute the shortest path from t to all network nodes.

```
Distance from  t  to  u is 2
Distance from  t  to  v is 4
Distance from  t  to  w is 4
Distance from  t  to  x is 7
Distance from  t  to  y is 7
Distance from  t  to  z is 15
Distance from  t  to  t is 0
```

b. Compute the shortest path from u to all network nodes.

```
Distance from  u  to  u is 0
Distance from  u  to  v is 3
Distance from  u  to  w is 2
Distance from  u  to  x is 6
Distance from  u  to  y is 9
Distance from  u  to  z is 14
Distance from  u  to  t is 2
```

c. Compute the shortest path from v to all network nodes.

```
Distance from  v  to  u is 3
Distance from  v  to  v is 0
Distance from  v  to  w is 4
Distance from  v  to  x is 3
Distance from  v  to  y is 8
Distance from  v  to  z is 11
Distance from  v  to  t is 4
```

d. Compute the shortest path from w to all network nodes.

```
Distance from  w  to  u is 2
Distance from  w  to  v is 4
Distance from  w  to  w is 0
Distance from  w  to  x is 6
Distance from  w  to  y is 11
Distance from  w  to  z is 14
Distance from  w  to  t is 4
```

e. Compute the shortest path from y to all network nodes.

```
Distance from  y  to  u is 9
Distance from  y  to  v is 8
Distance from  y  to  w is 11
Distance from  y  to  x is 6
Distance from  y  to  y is 0
Distance from  y  to  z is 12
Distance from  y  to  t is 7
```

f. Compute the shortest path from z to all network nodes.

```
Distance from  z  to  u is 14
Distance from  z  to  v is 11
Distance from  z  to  w is 14
Distance from  z  to  x is 8
Distance from  z  to  y is 12
Distance from  z  to  z is 0
Distance from  z  to  t is 15
```

Exercise 4 :

4.1 What values are returned during the following series of stack operations if executed upon an initially empty stack? push(5), push(3), pop(), push(2), push(8), pop(), pop(), push(9), push(1), pop(), push(7), push(6), pop(),pop(), push(4), pop(), pop().

The value returned will be [5].

```
[5, 3]
[5]
[5, 2, 8]
[5]
[5, 9, 1]
[5, 9]
[5, 9, 7, 6]
[5, 9]
[5, 9, 4]
[5]
```

4.2 Suppose an initially empty stack S has executed a total of 25 push operations, 12 top operations, and 10 pop operations, 3 of which raised Empty errors that were caught and ignored. What is the current size of S?

The current size of S is 15.

4.3 Implement a function with signature transfer(S, T) that transfers all elements from stack S onto stack T, so that the element that starts at the top of S is the first to be inserted onto T, and the element at the bottom of S ends up at the top of T.

```
40  S = [1,2,3,4,5]
41  T = []
42
43  for i in range(len(S)):
44      T.append(S[-1])
45      S.pop()
46
47  print(T)
```

Result :

```
[5, 4, 3, 2, 1]
```

Exercise 5 : What are network Intrusion detection, anomaly detection, malware detection, and misuse detection? Please explain their difference.

- A network intrusion detection system is a device or software application that monitors a network for malicious activity or policy violations. Any malicious activity or violation is typically reported or collected centrally using a security information and event management system.

- Anomaly detection is a step in data mining that identifies data points, events, and/or observations that deviate from a dataset's normal behavior.

- Malware Detection refers to a collection of techniques used to detect potentially harmful malware samples. These techniques are best employed as part of a robust defense system that works to detect malware samples before they have a chance to infect a victim’s system. However, this process can also take place after an infection has occurred.

- Misuse Detection is an approach in detecting potential insider threats to vulnerable company data, in which abnormal system behavior is defined first, and then any other behavior is defined as normal behavior.