Redes de computadoras, 28 de diciembre de 2024

Análisis del retardo de extremo a extremo

Martínez Buenrostro Jorge Rafael

correo, molap96@gmail.com
Universidad Autónoma Metropolitana
Unidad Iztapalapa, México

Procedimiento

Identificación de las trazas de audio

Para empezar se descargan las trazas y descomprimirlas utilizando los siguientes comando en la terminal de Linux

```
wget http://victor.ramos.online.fr/practica1/traces/1.txt.gz

gzip -dk 1.txt.gz
```

Figura 1: Comando para descargas y descomprimir las trazas

Para poder caracterizar el retardo de extremo a extremo, la Profra. Sue Moon propone añadir un elemento más a la diferencia; dicho elemento es la mínima diferencia de retardo encontrada en toda la traza. Esto significa encontrar $t_{min} = min\{t_i^r\} - t_i^t, \forall i$. Como las estampas de tiempo están codificadas con el protocolo RTP y han sido obtenidas con voz muestreada a 8,000 Hz, finalmente podremos observar el comportamiento del retardo de extremo a extremo dentro del paquete i con:

$$d^i_{end-to-end} = \frac{t_i^r - t_i^t - t_{min}}{8000}[seg.] \tag{1}$$

Para encontrar el *tiempo de sesión* para cada paquete, el análisis es idéntico al de una llamada telefónica tradicional, el que llama paga. Entonces, el tiempo de sesión estará basado en el reloj del transmisor. Este tiempo comenzará a correr a partir de la primera estampara de tiempo t_i^t hasta la última t_N^t dado que N es el número total de paquetes en la sesión. Entonces:

$$t^i_{session} = \frac{t_i^t - t_i^t}{8000}[seg] \tag{2}$$

Para analizar el retardo de extremo a extremo, entonces, se realizarán las gráficas de las Ecuaciones (1) y (2) para cada una de las seis trazas, a gran escala y a pequeña escala. La ecuación (1) es para el eje de las y y (2) para el eje de las x.

Manipulación de trazas con AWK

Para analizar algunas estadísticas básicas de las sesiones de VoIP capturadas en las trazas, se relizaron los siguientes scripts en AWK:

1. Script para contar el número total de frases (*talkspurts*) en una traza de retardo.

```
# Una frase terminal cuando se encuentra un periodo de silencio. Por lo tanto
# podemos contar los periodos de silencio como separadores de frases
BEGIN{
  talkspurts = 1
  input_file = ARGV[1] # Obtiene el nombre del archivo del parametro de la
      ejeucion
  gsub("../", "", input_file) # Elimina la parte inicial del parametro para
      tener nada mas el nombre del archivo
}

$1 == "!" {
  talkspurts++
}

END{
  print "Archivo de entrada:", input_file
  print "Numero total de frases:", talkspurts
  print ""
}
```

A continuación se muestra el comando que se usó para la ejecución del script. Ya que tenía que cumplir los siguientes requisitos para mantener el orden de las carpetas:

- La ubicación del script no es la misma que las trazas.
- La ubicación del script debía ser una ruta establecida.
- Ya que el resultado para cada traza será una línea, todas deben estar en el mismo archivo.

```
awk -f ./script1.awk ../1.txt > ./output
awk -f ./script1.awk ../6.txt >> ./output
```

2. Script para contar el número total de paquetes que llegaron al receptor.

```
# Cada paquete tiene una linea que comienza con "D".
BEGIN{
    paquetes = 0
    input_file = ARGV[1] # Obtiene el nombre del archivo del parametro de la
        ejeucion
    gsub("../", "", input_file) # Elimina la parte inicial del parametro para
         tener nada mas el nombre del archivo
}

$1 == "D"{
    paquetes++
}

END{
    print "Archivo de entrada:", input_file
    print "Numero total de paquetes recibidos:", paquetes
    print ""
}
```

Para la ejecución de este script usaremos como base la forma vista en el punto anterior. Cambiando el nombre del script de *script1.awk* a *script2.awk*

3. Script para encontrar la diferencia mínima entre la estampa de tiempo de receptor menos de la del emisor

```
# Este script encuentra la diferencia minima entre la estampa del receptor y
    la del emisor.
BEGIN{
    min_diff = "Inf"
    input_file = ARGV[1] # Obtiene el nombre del archivo del parametro de la
        ejeucion
    gsub("../", "", input_file) # Elimina la parte inicial del parametro para
         tener nada mas el nombre del archivo
}

$1 == "D"{ # Solo se calcula la diferencia cuando hay un paquete recibido
    diff = $2 - $3 # Ya que solo es la diferencia no se toma encuenta que el
        numero que se obtiene puede ser negativo
    if (diff < min_diff){
        min_diff = diff
    }
}

END{
    print "Archivo de entrada:", input_file
    print "Diferencia minima:", min_diff
    print ""
}
```

Para la ejecución de este script usaremos como base la forma vista en el punto anterior. Cambiando el nombre del script de *script2.awk* a *script3.awk*

4. Script que recibe una traza como entrada y entrega en un archivo dos columnas: tiempo de la sesión en segundos, y retardo de extremo a extremo a extremo en segundo.

```
# Este script genera un archivo con dos columnas: tiempo de la sesion y
    retardo extremo a extremo.
BEGIN{
    #Llama al script que se encarga de obtener la diferencia minima
    temp_file = "output.tmp"
    input_file = ARGV[1]
    command = "awk -f ../Script_3/script3.awk "input_file " > " temp_file
    system(command)

    # Procesa la salida del archivo temporal
    while ((getline line < temp_file) > 0) {
        # Procesa cada linea de salida del segundo script
        if (line ~ /Diferencia minima:/) {
            split(line, fields, ":")
            tmin = fields[2]
        }
    }

    # Limpia el archivo temporal
    system("rm -f " temp_file)

    t1_flag = 0
}

$1 == "D"{
    if (t1_flag == 0){
        t1 = $3
        t1_flag = 1
    }

    session_time = ($3 - t1) / 8000
    end_to_end_delay = ($2 - $3 - tmin) / 8000
    print session_time, end_to_end_delay
}
```

A continuación se muestra el comando que se usó para la ejecución del script. Ya que se tiene que generar un archivo para cada una de las trazas

```
awk -f ./script1.awk ../1.txt > ./output
```

5. Script que calula el retardo promedio de extremo a extremo en una sesión

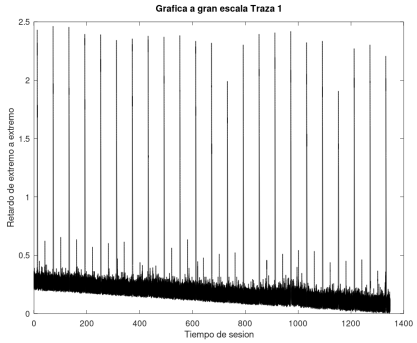
```
# Este script calcula el retardo promedio extremo a extremo.
BEGIN{
    tmin = "Inf" # Inicializa con el valor infinito
    total_delay = 0
    input_file = ARGV[1]
    gsub("../", "", input_file)
}

$1 == "D"{

    if ($2-$3 < tmin) {
        tmin = $2-$3
    }

    delay = ($2 - $3 - tmin) / 8000
    total_delay += delay
    count++
}

END{
    print "Archivo de entrada:", input_file
    if (count > 0) {
        print "Retardo promedio extremo a extremo:", total_delay / count
    } else {
        print "No se encontraron datos para calcular el retardo promedio."
    }
    print ""
}
```

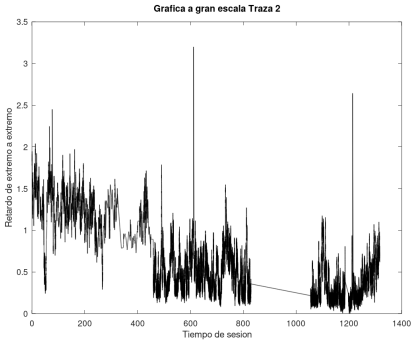
Para la ejecución de este script usaremos como base la forma vista en el primer punto. Cambiando el nombre del script de *script1.awk* a *script5.awk*

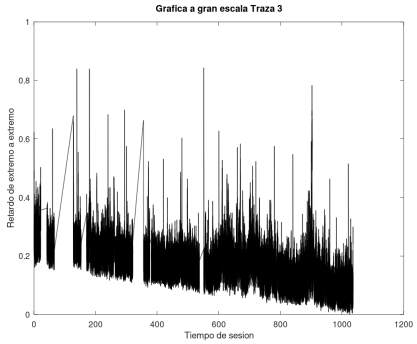
Comportamiento

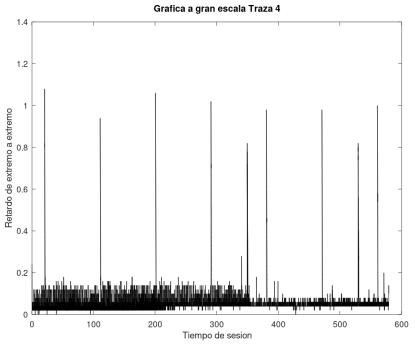
1. Gráficas a gran escala. Se trazan todos los elementos del retardo de extremo a extremo. Para trazar estas gráficas usaremos las siguientes líneas en Octave

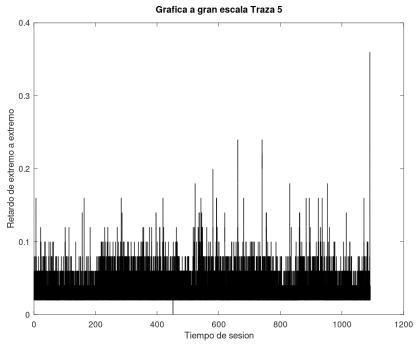
```
data = load('output1.txt');
x = data(:, 1);
y = data(:, 2);
plot (x,y, '-o', 'Color', 'k');
xlabel ('Tiempo de sesion');
ylabel ('Retardo de extremo a extremo');
title ('Grafica a gran escala Traza 1');
print -dpng "../../Reporte/img/traza1_GE.png";
```

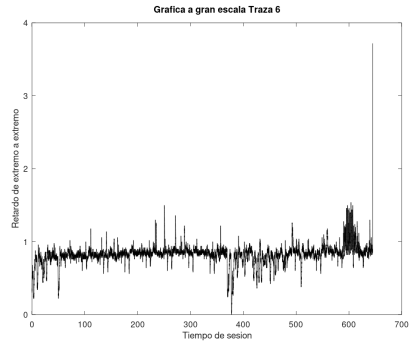










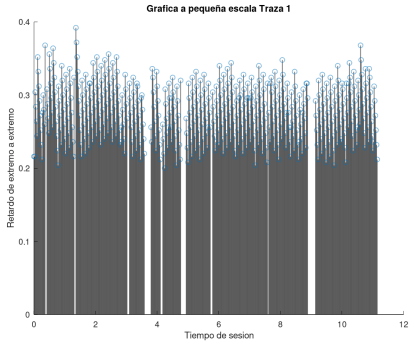


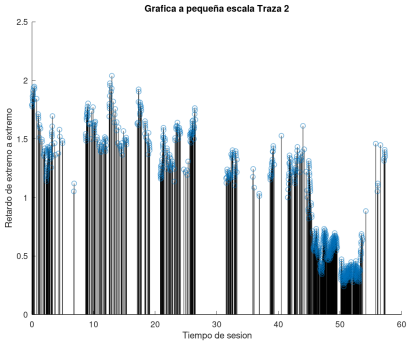
2. Gráficas a pequeña escala. Se trazan dos o tres frases en la conversación. El primer paso para poder trazar las gráficas a pequeña escala es acotar las sesiones a unas cuantas frases. Para esto usaremos el siguiente script en AWK que solo obtiene los datos de las primeras cuatro frases

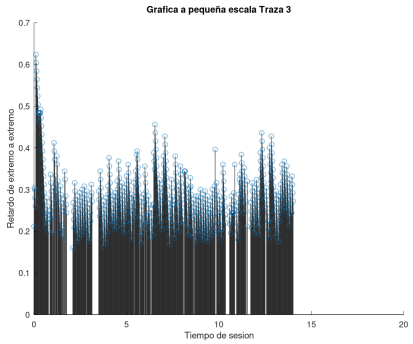
```
# Este script genera un archivo con dos columnas: tiempo de la sesion y
    retardo extremo a extremo.
BEGIN{
    #Llama al script que se encarga de obtener la diferencia minima
    temp_file = "output.tmp"
    input_file = ARGV[1]
    command = "awk -f ../Script_3/script3.awk "input_file " > " temp_file
    system(command)

    # Procesa la salida del archivo temporal
    while ((getline line < temp_file) > 0) {
        # Procesa cada linea de salida del segundo script
        if (line ~ /Diferencia minima:/) {
            split(line, fields, ":")
            tmin = fields[2]
        }
    }

    # Limpia el archivo temporal
    system("rm -f " temp_file)

    t1_flag = 0
    talkspurts = 1
}

$1 == "D"{

    if (t1_flag == 0){
        t1 = $3
        t1_flag = 1
    }

    session_time = ($3 - t1) / 8000
    end_to_end_delay = ($2 - $3 - tmin) / 8000
    print session_time, end_to_end_delay
}


$1 == "!"{
  talkspurts++ # Incrementa el contador al encontrar un espacio de silencio
  if(talkspurts == 11){
    exit # Termina la ejecucion si encuentra cuatro signos "!"
  }
}
```

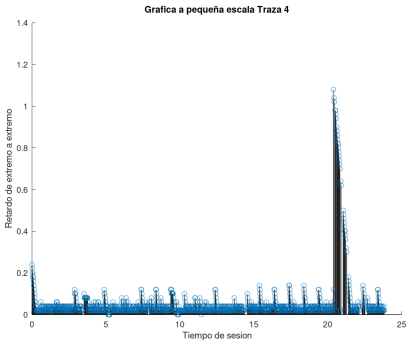
El siguiente paso es usar el siguiente script en Octave para poder trazar las gráficas

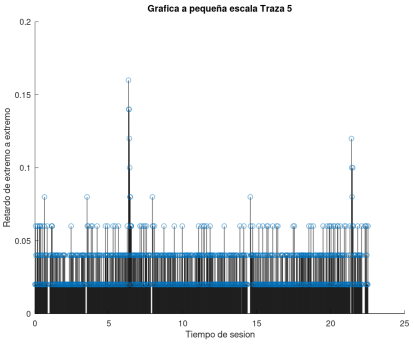
```
data = load('output1_PE.txt');
x = data(:, 1);
y = data(:, 2);
stem (x,y, 'Color', 'k');
xlabel ('Tiempo de sesion');
ylabel ('Retardo de extremo a extremo');
title ('Grafica a gran escala Traza 1');
print -dpng "../../Reporte/img/traza1_PE.png";
```

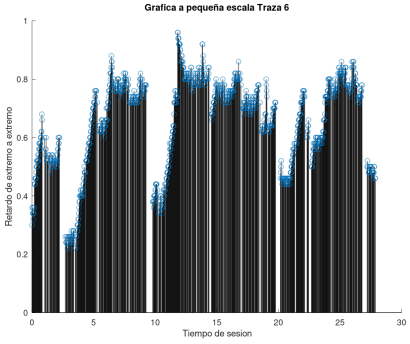












3. ¿Cúales son los fenómenos particulares que observa usted en las gráficas que acaba de obtener?

4. Tabla con las estadísticas obtenidas

# Traza	Total de frases	Total de paquetes	Diferencia mínima	Retardo promedio extremo a extremo
1	818	56,979	-641,808	0.110536
2	406	24,490	-14,643,186	0.392336
3	536	37,640	-774,206	0.109097
4	252	27,814	-807,010	0.0486863
5	540	52,836	-944,917	0.0224502
6	299	23,293	-1,489,551	0.690046

5.