

## Tarea 1 Metaheurísticas. Evaluador AFP y Script de ejecuciones.

**Descripción del método.** El algoritmo para el evaluador de soluciones al FS-FAP esta basado en el artículo “Path relinking for the fixed spectrum frequency assignment problem”, el cual plantea una situación en la que se tiene un numero finito de transmisores a los cuales se les asigna una determinada frecuencia, y la interferencia entre dos transmisores esta determinada por una variable de peso y una penalización que se aplica cuando la distancia en frecuencia entre dos transmisores es inferior al peso dado; por tanto la problemática consiste en determinar el mapeo (una asignación de frecuencia para cada transmisor) que produzca la mínima penalización.

### Programa Evaluador.

Consiste esencialmente en una función que recibe el numero de frecuencias a asignar y el numero de transmisores, y efectúa una asignación aleatoria para cada transmisor, utilizando la función `rand()` de C, así como otra función que a partir de 1000 soluciones generadas calcula el costo total de cada solución generado por las penalizaciones, conservando el mejor mapeo y el costo minimo, los datos del peso y penalizaciones para cada par de frecuencias fueron tomados de las 9 instancias GSM2-XXX propuestas en el articulo:

Graph	F
GSM2-184	39
GSM2-184	49
GSM2-184	52
GSM2-227	29
GSM2-227	39
GSM2-227	49
GSM2-272	34
GSM2-272	39
GSM2-272	49

**Compilación.** El programa compila con `g++`. Archivo Makefile incluido, el comando `make` compila los ficheros `main.cpp` y la librería `FSFAP.hpp` contenidas en la carpeta “src”, y genera los objetos en la carpeta “obj” y el ejecutable en “bin”.

```
almeida@almeida-X501A1:~$ cd /home/almeida/Dropbox/Metaheurísticas/EvaluadorFSFAP/EvaluadorFSFAP
almeida@almeida-X501A1:~/Dropbox/Metaheurísticas/EvaluadorFSFAP/EvaluadorFSFAP$ make
g++ -std=c++11 -c src/FSFAP.cpp -o obj/FSFAP.o
g++ -std=c++11 -c src/main.cpp -o obj/main.o
g++ -std=c++11 -o bin/ejecutable obj/FSFAP.o obj/main.o
```

**Ejecución.** `make run` ejecuta el programa, se utiliza la directiva `args` para pasar los argumentos:

```
almeida@almeida-X501A1:~/Dropbox/Metaheurísticas/EvaluadorFSFAP/EvaluadorFSFAP$
make run args="GSM2-184.ctr 184 39 Results.dat"
./bin/ejecutable GSM2-184.ctr 184 39 Results.dat
```

**Resultados.** Debido a que la soluciones generadas son puramente aleatorias, no se esta aplicando ningún tipo de heurística en lo que respecta a la asignación de frecuencias, por lo tanto los resultados obtenidos son bastante malos, el programa anexa al archivo “Results.dat” los resultados que se van obteniendo.



```
Abrir  Results 2.dat  /tmp/fz3temp-2
184 39 3700331386
184 39 4300262990
184 39 3800281536
184 39 3900247675
184 39 3900277558
184 39 4300246523
184 39 4400235829
184 39 3700258752
184 39 4100266478
Anchura de la pestaña: 8  Ln 10, Col 18  INS
```

Parte del archivo resultados generado. Por columnas se muestra el numero de transmisores, frecuencias y mínimo costo obtenido

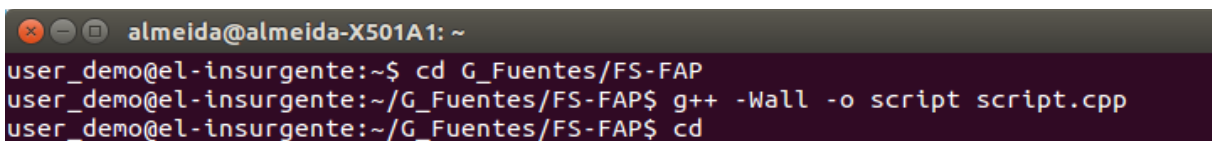
### Script de Ejecuciones.

Consiste en un programa que recibe de un archivo una lista con las identificaciones de cada uno de las nodos que utilizaremos para hacer las ejecuciones (machinefile.txt) y de otro archivo las directivas con las que correremos 100 veces cada una de las instancias de nuestro algoritmo (taskfile.txt). El programa esta codificado en C++, utilizando objetos de la librería estándar STL, y utilizando las funciones fork() y wait() de la librería unistd.h de C, así como el comando system() para ejecutar las instancias. El algoritmo consiste básicamente en un ciclo de forks donde a cada hijo se le da un comando para ejecutar, luego el programa espera a que cada uno de los hilos termine su proceso para lanzar un nuevo ciclo de hilos. El comando a ejecutar tiene la siguiente estructura:

ssh + [ID del nodo] + [directorio del ejecutable] + [argumentos]....

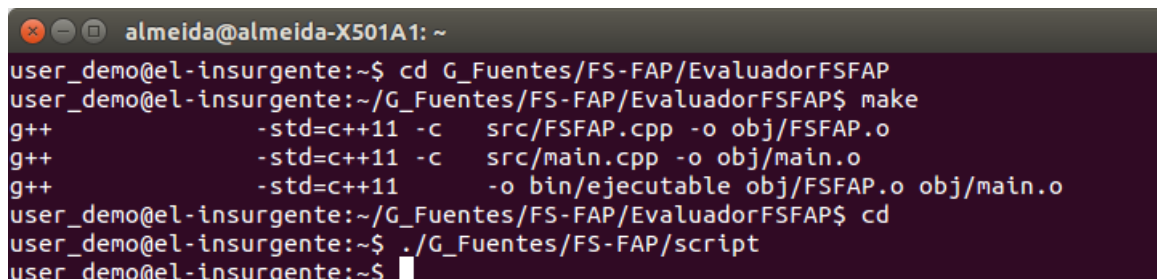
El ID del nodo es tomado del archivo machinefile.txt, el ejecutable en este caso es nuestro evaluador, y los argumentos para cada ejecución son tomados del archivo taskfile.txt.

**Compilación/Ejecución.** El script compila con g++, no necesita argumentos o librerías adicionales, se coloca en la misma carpeta que el fichero que contiene el evaluador (fuera de este).



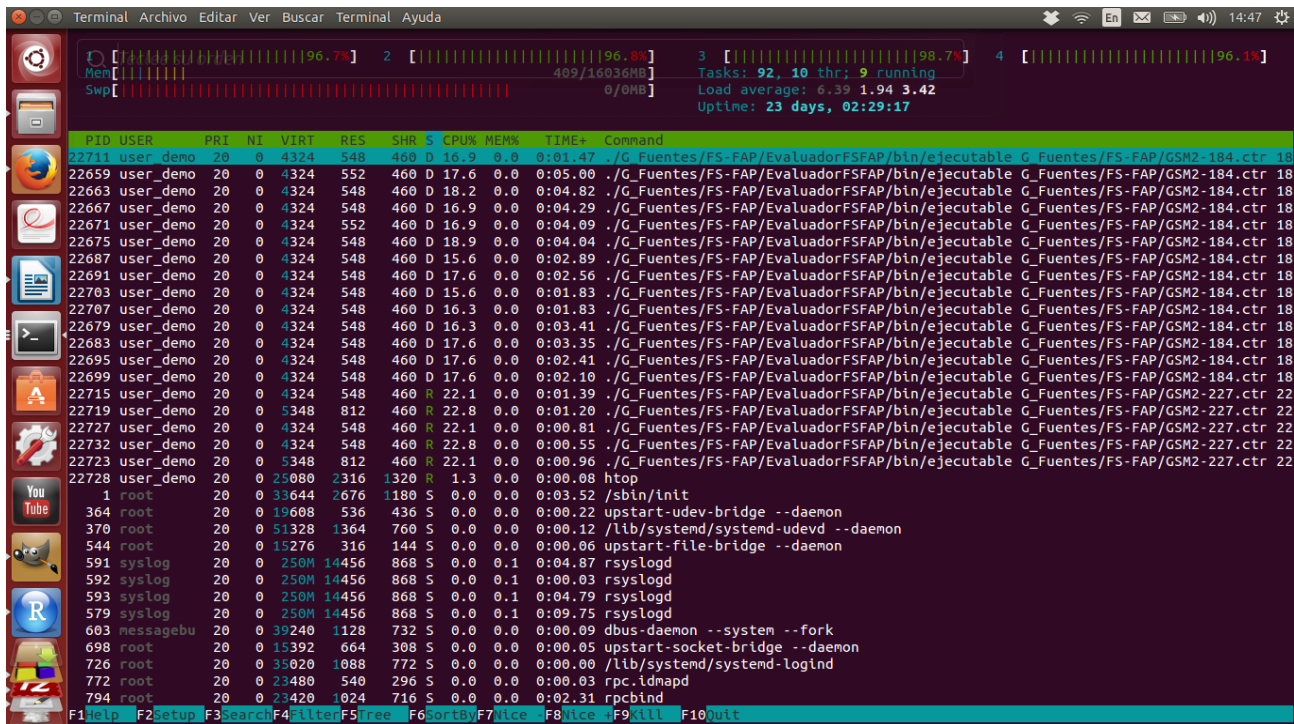
```
almeida@almeida-X501A1: ~
user_demo@el-insurgente:~$ cd G_Fuentes/FS-FAP
user_demo@el-insurgente:~/G_Fuentes/FS-FAP$ g++ -Wall -o script script.cpp
user_demo@el-insurgente:~/G_Fuentes/FS-FAP$ cd
```

**Resultados.** El archivo de tareas contiene 100 directivas para cada una de las 9 instancias que se va a ejecutar en el cluster, cada instancia ejecutada anexa su resultado al archivo “Results.dat”. Para este ejemplo utilizamos los nodos c-0-10 al c-0-19 del cluster EL Insurgente.

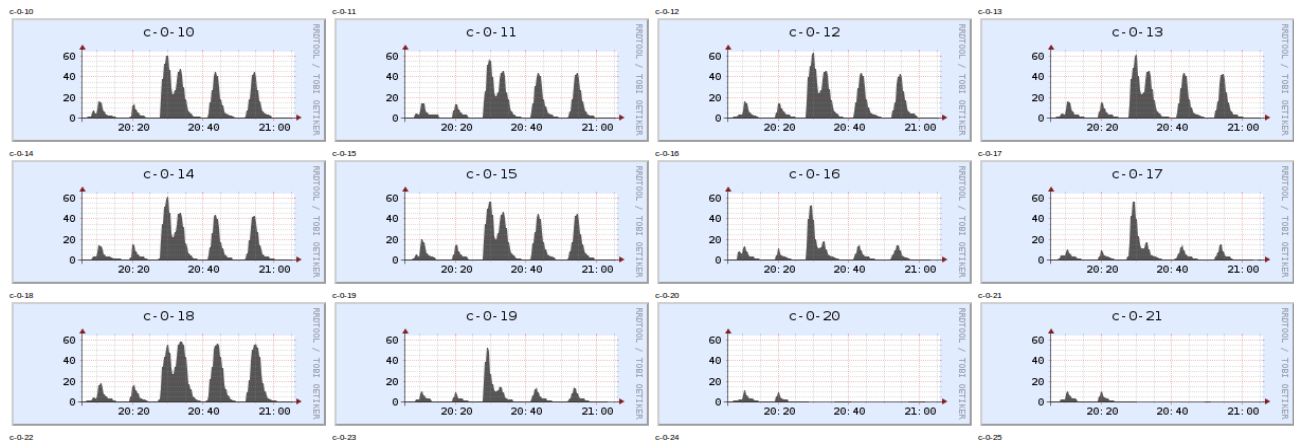


```
almeida@almeida-X501A1: ~
user_demo@el-insurgente:~$ cd G_Fuentes/FS-FAP/EvaluadorFSFAP
user_demo@el-insurgente:~/G_Fuentes/FS-FAP/EvaluadorFSFAP$ make
g++ -std=c++11 -c src/FSFAP.cpp -o obj/FSFAP.o
g++ -std=c++11 -c src/main.cpp -o obj/main.o
g++ -std=c++11 -o bin/ejecutable obj/FSFAP.o obj/main.o
user_demo@el-insurgente:~/G_Fuentes/FS-FAP/EvaluadorFSFAP$ cd
user_demo@el-insurgente:~$ ./G_Fuentes/FS-FAP/script
user_demo@el-insurgente:~$
```

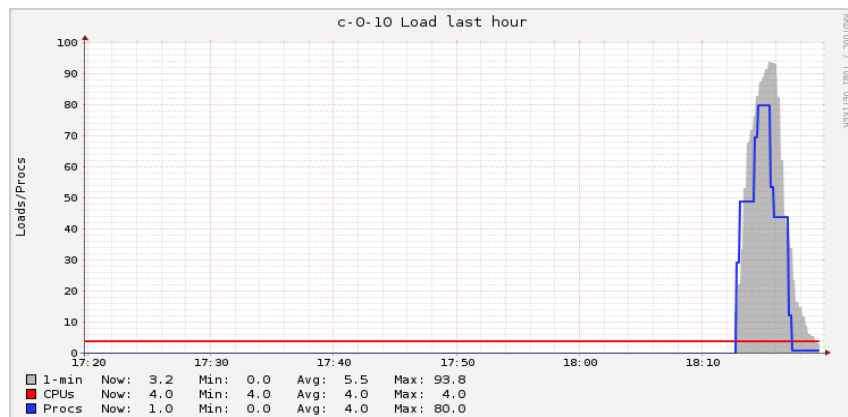
Ejecución del script



Nodo C-0-10 ejecutando varias instancias



Actividad en cada uno de los Nodos



Carga en el Nodo C-0-10 después de ejecutar el script

A continuación se muestra una tabla comparativa de los resultados que se obtuvieron. Se observa que para cada caso de un mismo numero de transistores, el Costo mínimo promedio va disminuyendo conforma va aumentando el numero de frecuencias, a su vez, el Costo mínimo promedio aumenta al aumentar el numero de transmisores, máximos y mínimos para cada caso se muestran como referencia.

Num de Transmisores	Num de Frecuencias	Costo promedio	Máximo	Mínimo
184	39	4050775674	4500277213	3400252326
184	49	3025213133	3400244386	2300216954
184	52	2797208394	3100240732	2400173618
227	39	6307488226	6700350581	5500390497
227	49	4730334916	6800373217	5300522862
227	52	4407445252	4900279024	3500219618
272	39	8135951135	8700539677	7100593702
272	49	6194347191	6800373217	5300522862
272	52	5788449759	6200471373	4900420260

**Nota:** Aunque se pudo solucionar el problema de conexión colocando un retardo (0.2 seg) antes de llamar a la función fork(), observe en todos los casos que el cluster no me devolvía los 900 resultados , faltando en promedio un 1% de estos, probablemente debido a la rapidez con que se ejecutan las tareas o a que el retardo debiera ser un poco mas prolongado aunque sacrificando un poco el tiempo de ejecución, se espera que conforme se avance durante el curso se depuren este tipo de errores.

Los archivos de programas vienen anexos con el reporte, pero también se encuentran en la carpeta ***home/user\_demo/G\_Fuentes/FS-FAP*** de donde se ejecutaron en el cluster.