Tarea 4 Metaheurísticas: Análisis de Resultados.

Introducción.

En esta práctica el objetivo es realizar una comparativa entre 17 configuraciones de Metaheurísticas que fueron implementadas y aplicadas al Problema de Asignación de Frecuencias (AFP) en la ciudad de Denver, Colorado. Para cada configuración se realizaron 30 ejecuciones con una duración de 48 hrs. cada una, registrando el estado de la población cada 10 minutos.

La comparativa se realizó en base a un análisis cuantitativo y cualitativo que consistió en las siguientes validaciones experimentales:

- Análisis Estadístico de los mejores resultados
- Tabla de métricos estadísticos
- Box-plots de calidad por algoritmo
- Gráficas de evolución (media, mediana, máximo, mínimo, desviación estándar)
- Evolución de la diversidad (Entropía, distancia de Hamming)
- Run Length Distribution (RLD)

Los datos que sirvieron de base para este análisis fueron obtenidos a partir de 30 archivos de texto para cada algoritmo, en los que se utilizó un programa en C++ (estándar C++11) para generar 17 archivos que resumen toda la información necesaria para elaborar cada una de las gráficas, las cuales fueron procesadas en el software de RStudio versión 3.2.3 con la librería ggplot2.

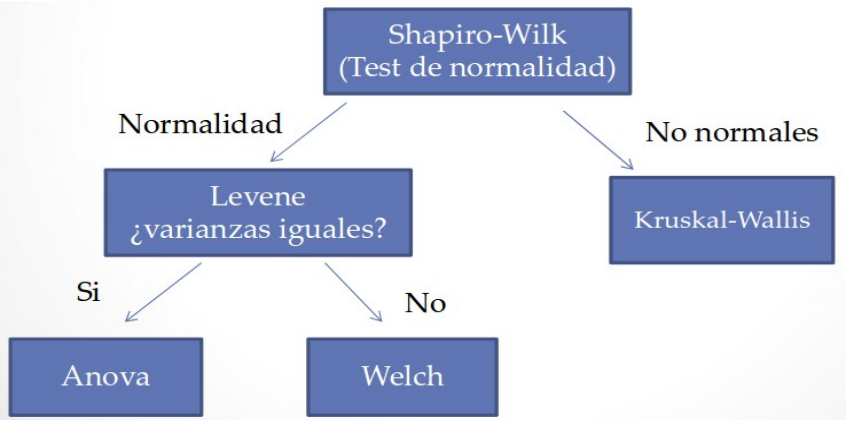
Algoritmos que fueron analizados durante esta práctica:

ALGORITMO	Abreviación
CDRWGenerationalParentBinaryTournament	CDRW
ClearingParentBinaryTournamentElitistWinnersAboveAverage_1	CLR_1
ClearingParentBinaryTournamentElitistWinnersAboveAverage_2	CLR_2
ClearingParentBinaryTournamentElitistWinnersAboveAverage_5	CLR_5
CrowdingMahfoudA5	A5
CrowdingMahfoudA5ScaledProbabilistic	A5_scaled
CrowdingRestrictedTournamentSelectionParentBinaryTournament_2	RTS_2
CrowdingRestrictedTournamentSelectionParentBinaryTournament_5	RTS_5
CrowdingRestrictedTournamentSelectionParentBinaryTournament_10	RTS_10
CrowdingRestrictedTournamentSelectionParentBinaryTournament_25	RTS_25
CrowdingRestrictedTournamentSelectionParentBinaryTournament_50	RTS_50
DiversityAdaptiveGeneralizedCrowding_0.25	AGCR_025
DiversityAdaptiveGeneralizedCrowding_0.75	AGCR_075
GenerationalElit	GEN_ELIT
ObjectivesCombinedParentBinaryTournament	COMB
ReplaceWorstFromParentAndOffspring	RW
MultiObjNearestIndDistThresholdPopPercentReference	MULTY_DYNAMIC

i) Análisis Estadístico.

Se siguió el análisis estadístico revisado en clase y propuesto en el artículo *"A Novel Diversity-Based Replacement Strategy for Evolutiory Algorithms"*. Tal análisis consiste en que para un nivel de significancia del 5%, se realiza una prueba de **Shapiro-Wilk** para cada conjunto de resultados con el fin de verificar si los datos provienen de una distribución Gaussiana, de ser así, se realiza la prueba de **Levene** para verificar homogeneidad de varianzas, si las muestras tienen varianzas iguales, se realiza una prueba **ANOVA**, si las varianzas son diferentes, se efectúa una prueba de **Welch**. Para muestras que resulten no provenir de distribuciones Gaussianas, se realiza la prueba no paramétrica de **Kruskal-Wallis** para verificar si las muestras provienen de una misma distribución.

Se sigue también el criterio propuesto en el artículo antes mencionado para decidir cuando un algoritmo es mejor que otro; se tiene en cuenta que si el test estadístico correspondiente nos arroja un *pvalue* menor al 5%, se concluye que las diferencias entre las muestras que se están probando son estadísticamente significantes y el algoritmo ganador lo decide la mejor media en el caso de **ANOVA** y **Welch**, o la mejor mediana en el caso de **Kruskal-Wallis**.



Cabe señalar que no siempre las muestras que se tomaron resultaron ser comparables, o que no siempre los test que se efectuaron mostraron diferencias significativas. A continuación se muestra un resumen de los resultados de las pruebas realizadas para cada par de algoritmos, se toma como muestra el mejor valor de *fitness* obtenido para cada una de las 30 ejecuciones, y para cada columna un valor de 1 significa que el algoritmo resulto ser mejor que el algoritmo del renglón correspondiente, un -1 indica que fue inferior y un 0 indica que no se pudo realizar una comparación:

	CDRW	CLR_1	CLR_2	CLR_5	A5	A5_scaled	RTS_2	RTS_5	RTS_10	RTS_25	RTS_50	AGCR_025	AGCR_075	GEN_ELIT	COMB	RW	MULTY_DYNAMIC
CDRW	0	0	-1	-1	1	1	-1	-1	0	1	1	-1	-1	0	1	0	1
CLR_1	0	0	-1	-1	1	1	-1	-1	0	1	1	-1	-1	0	1	0	1
CLR_2	1	1	0	0	1	1	-1	0	0	1	1	0	-1	0	1	0	1
CLR_5	1	1	0	0	1	1	-1	0	0	1	1	0	-1	0	1	0	1
A5	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	1
A5_scaled	-1	-1	-1	-1	1	0	-1	-1	-1	-1	0	-1	-1	-1	0	-1	1
RTS_2	1	1	1	1	1	1	0	1	1	1	1	1	-1	1	1	1	1
RTS_5	1	1	0	0	1	1	-1	0	0	1	1	0	-1	0	1	0	1
RTS_10	0	0	0	0	1	1	-1	0	0	1	1	0	-1	0	1	0	1
RTS_25	-1	-1	-1	-1	1	1	-1	-1	-1	0	1	-1	-1	-1	0	-1	1
RTS_50	-1	-1	-1	-1	0	0	-1	-1	-1	-1	0	-1	-1	-1	-1	-1	1
AGCR_025	1	1	0	0	1	1	-1	0	0	1	1	0	-1	0	1	0	1
AGCR_075	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
GEN_ELIT	0	0	0	0	1	1	-1	0	0	1	1	0	-1	0	1	0	1
COMB	-1	-1	-1	-1	1	0	-1	-1	-1	0	1	-1	-1	-1	0	-1	1
RW	0	0	0	0	1	1	-1	0	0	1	1	0	-1	0	1	0	1
MULTY_DYNAMIC	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

La siguiente tabla muestra un resumen de los resultados que se obtuvieron al hacer el “torneo” entre todos los algoritmos, la flecha hacia arriba indica las veces que el algoritmo gano y la flecha hacia abajo indica las veces que perdió:

Strategy	↑	↓	Score
CDRW	6	6	0
CLR_1	6	6	0
CLR_2	2	8	-6
CLR_5	2	8	-6
A5	14	1	13
A5_scaled	12	2	10
RTS_2	1	15	-14
RTS_5	2	8	-6
RTS_10	2	6	-4
RTS_25	11	4	7
RTS_50	13	1	12
AGCR_025	2	8	-6
AGCR_075	0	16	-16
GEN_ELIT	2	6	-4
COMB	11	3	8
RW	2	6	-4
MULTY_DYNAMIC	16	0	16

Claramente se aprecia que el algoritmo MULTY_DYNAMIC resultó ganador en todas las ocasiones y por consiguiente fue el mejor algoritmo según este test estadístico, seguido por el algoritmo *Crowding Mahfoud* A5 y el *Crowding Restricted Tournament Selection Parent Binary Tournament_50* (RTS_50), en segundo y tercer lugar respectivamente.

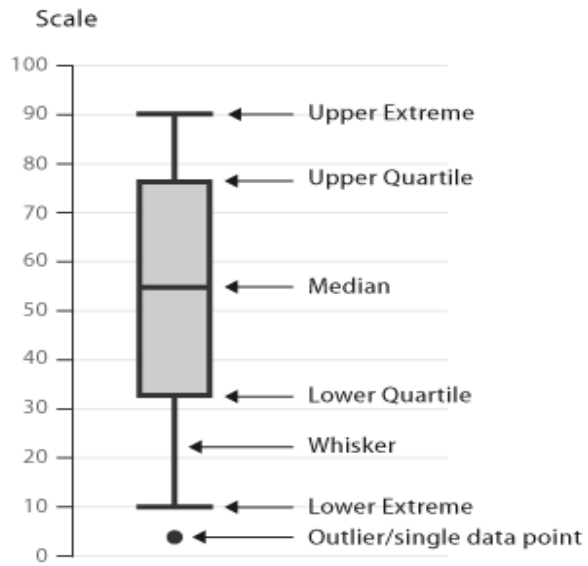
ii) Tabla de métricos estadísticos.

La siguiente tabla muestra las medidas estadísticas para cada conjunto de resultados que se analizaron, se observa que el algoritmo MULTY_DYNAMIC presenta los mejores valores en media, mediana, máximo y mínimo de todo el conjunto, se observa en segundo y tercer lugar nuevamente, los algoritmos que ya habíamos mencionado con anterioridad.

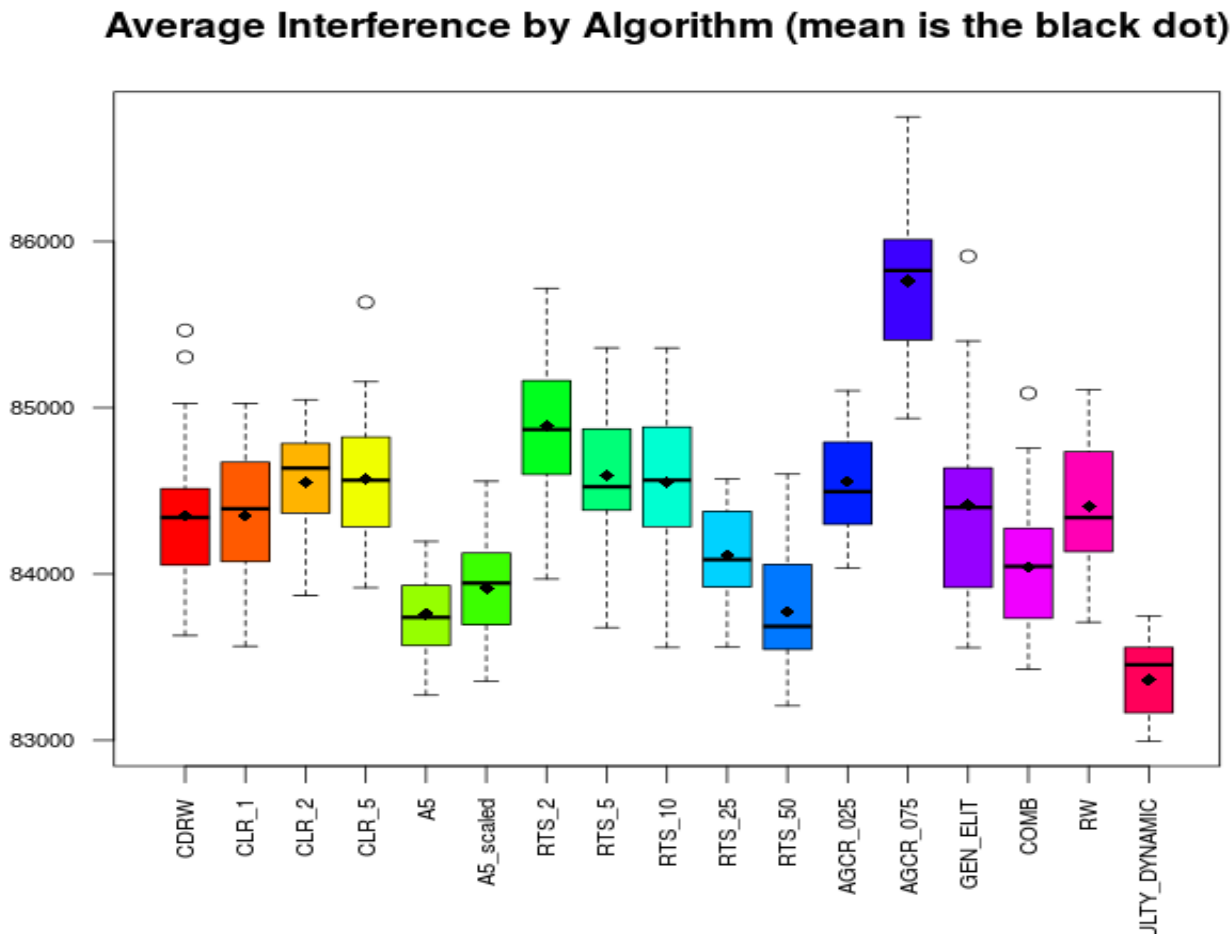
Algorithm	Mean	Median	StdDev	Max	Min
CDRW	84348.84	84340.04	445.3958	85465.6	83630.04
CLR_1	84350.81	84392.02	382.4131	85026.7	83565.39
CLR_2	84554.17	84636.63	337.1876	85048.27	83871.25
CLR_5	84571.71	84563.71	410.4391	85634.6	83917.68
A5	83760.04	83739	256.9934	84196.06	83270.84
A5_scaled	83913.26	83945.67	316.2129	84557.74	83354.05
RTS_2	84893.47	84868.61	448.3392	85717.84	83969.9
RTS_5	84591.01	84524.68	423.0613	85359.58	83677.53
RTS_10	84550.04	84564.26	403.695	85358.96	83558.24
RTS_25	84115.54	84086.56	272.2805	84571.28	83559.66
RTS_50	83770.98	83685.2	348.3665	84600.62	83207.79
AGCR_025	84557.22	84495.08	295.3535	85102.31	84034.49
AGCR_075	85762.81	85825.14	434.1341	86749.25	84936.14
GEN_ELIT	84415.23	84400.09	566.6219	85911.18	83556.96
COMB	84043.45	84045.94	383.0702	85087.37	83427.24
RW	84407.4	84339.15	360.2388	85108.49	83709.79
MULTY_DYNAMIC	83361.81	83454.01	242.0886	83747.25	82994.93

iii) Box-plots de Calidad.

Los Box-plots son una herramienta de visualización que resulta muy útil para representar algunas propiedades importantes de una población, tales como la mediana, cuartiles, máximos, mínimos y datos atípicos.



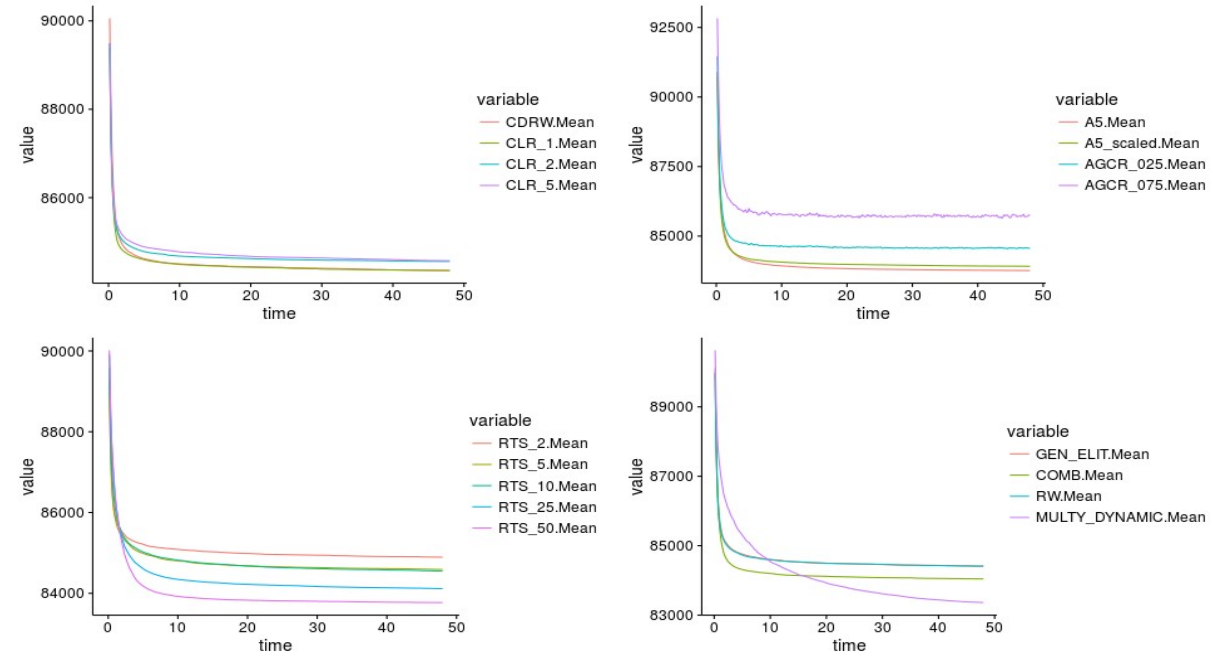
La siguiente gráfica muestra las distintas poblaciones para cada uno de los algoritmos analizados en esta práctica, se toman como muestra los mismos datos que utilizamos para realizar nuestro test estadístico. Aunque se observa que muchas de las medias en cada uno de los box-plots son diferentes entre sí, también se observa diferentes niveles de variación entre cada población (largo del boxplot) y se observan muchos traslapes entre todos esos valores, por eso resulta necesario el análisis estadístico efectuado anteriormente para descartar que estas diferencias hayan ocurrido sólo por azar.



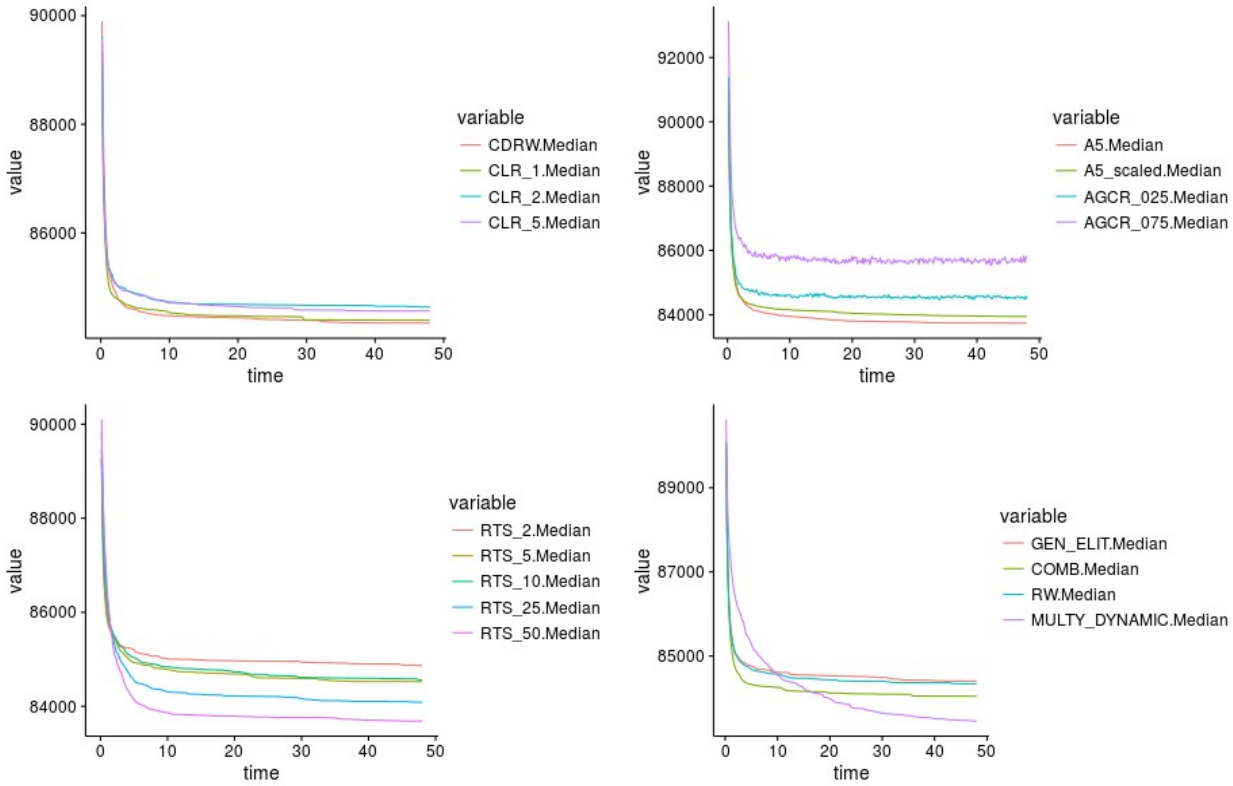
iv) Gráficas de Evolución.

Las gráficas de Evolución nos muestran el comportamiento del algoritmo conforme el tiempo avanza y se van generando cada vez nuevas poblaciones e individuos. En nuestro caso la función de fitness, que es la función que guía esta evolución, esta relacionada con las penalizaciones ocasionadas por las interferencias entre los transmisores, y conforme el algoritmo avanza se busca minimizar el resultado de esta función.

Para decidir que mediciones vamos a monitorizar, se debe tomar en cuenta lo que queremos demostrar con nuestros datos, ya sea en cuestión de calidad, diversidad, peores o mejores soluciones, poblaciones, etc., y así establecer pautas que nos permitan diferenciar un algoritmo de otro en términos de robustez, eficiencia y calidad de soluciones. Para nuestro caso en particular mostramos gráficas de medias, medianas, máximos y mínimos para cada uno de los algoritmos en cuestión, se muestran diferentes gráficas para cada métrica para facilitar la visualización; se muestra la evolución del mejor valor de fitness en cada momento en función del tiempo expresado en horas .

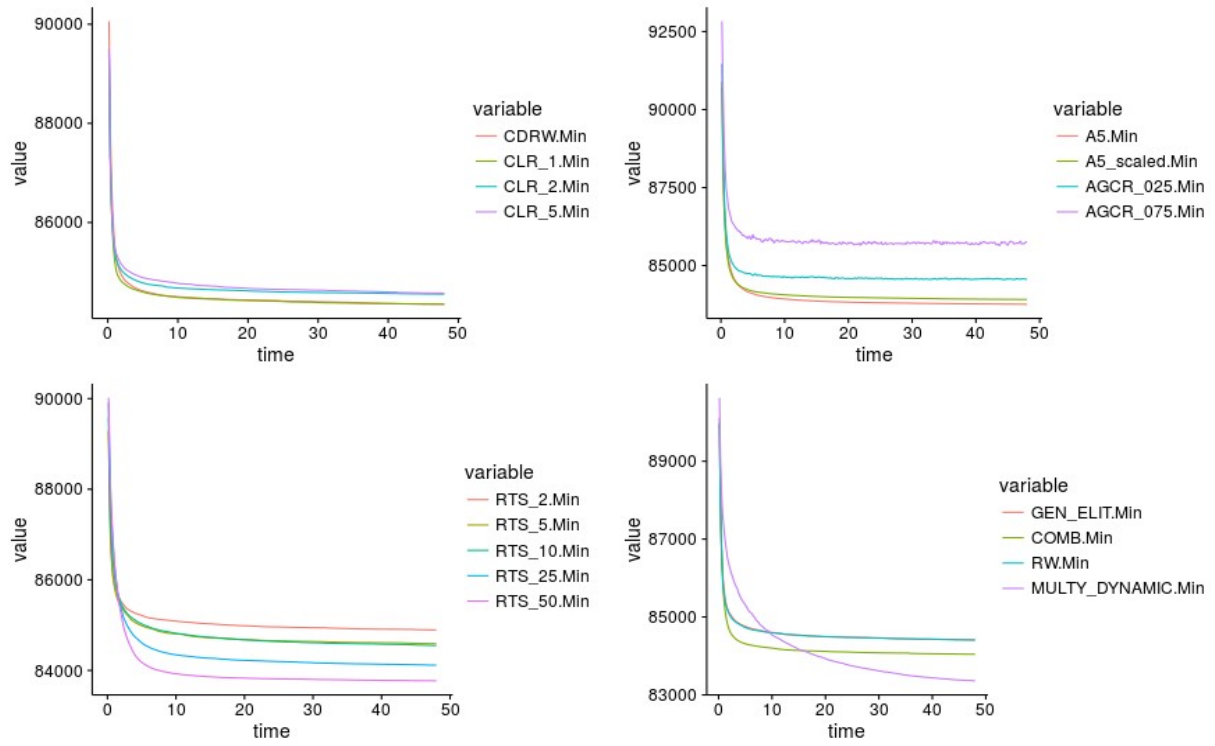


Gráfica 1. Evolución de la media de los mejores fitness para cada población

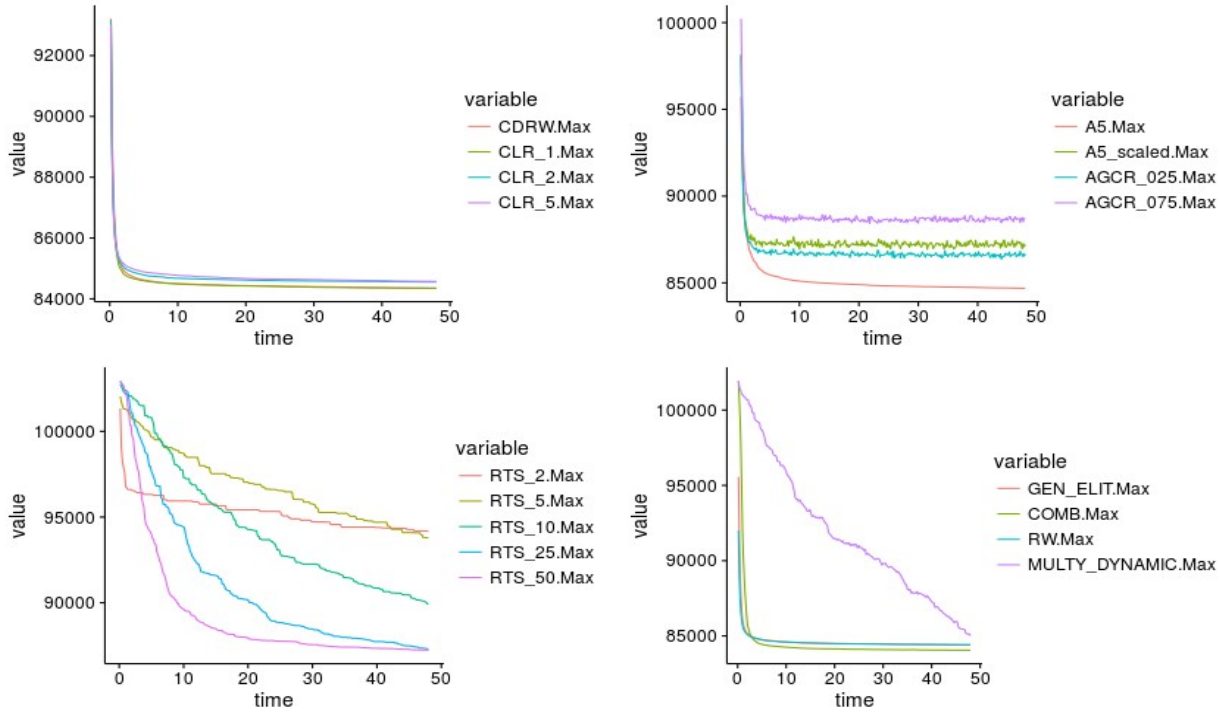


Gráfica 2. Evolución de la mediana de los mejores fitness para cada población

Como se esperaba, se observa un comportamiento similar entre las gráficas de evolución de la media y la mediana, si echamos un vistazo a la gráfica de box-plots, nos daremos cuenta que en la mayoría de los casos, la media (representada con un punto) se encuentra bastante cerca de la línea de la mediana.



Gráfica 3. Evolución de los valores mínimos de los mejores fitness para cada población



Gráfica 4. Evolución de los valores máximos de los mejores fitness para cada población

En la gráfica de mínimos, se observa en el algoritmo MULTY_DYNAMIC es el que mas desciende, obteniendo los mejores resultados de toda la prueba, mientras que los demás convergen rápidamente a un óptimo local y ya no descienden mas, incluso el comportamiento del máximo es similar, con excepción de los algoritmos RTS y el MULTY_DYNAMIC, que presentan variaciones mas graduales.

v) Evolución de la Diversidad.

Se entiende por Diversidad a la variabilidad entre las soluciones que un algoritmo genera conforme se acerca a una solución que produzca un valor de *fitness* óptimo. Las métricas que se utilizaron en esta practica para medir la diversidad consisten en la Entropía poblacional y la media de la distancia al individuo mas cercano.

Entropía Poblacional.

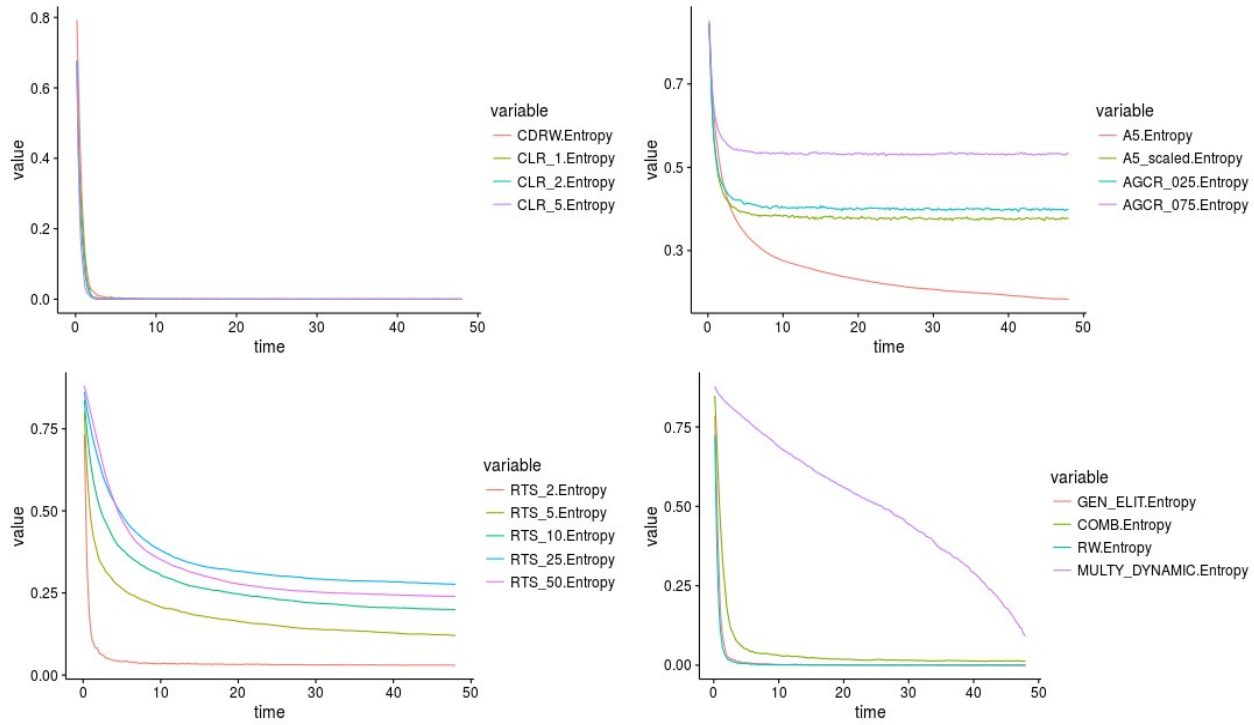
Mientras mas variabilidad haya entre los individuos de una población, la Entropía Poblacional sera elevada:

$$H(t)=\frac{1}{n}\sum_{i=1}^{n} H_i(t) \quad \text{donde} \quad H_i(t)=-\sum_{j=1}^{n} P_{ij}\log_{v_i}(P_{ij})$$

$H_i(t)$=Entropia del gen i
v_i= número de posibles valores para el gen i
P_{ij}=frecuencia de los individuos en la población cuyo gen i toma el j-*é*simo valor posible para ese gen

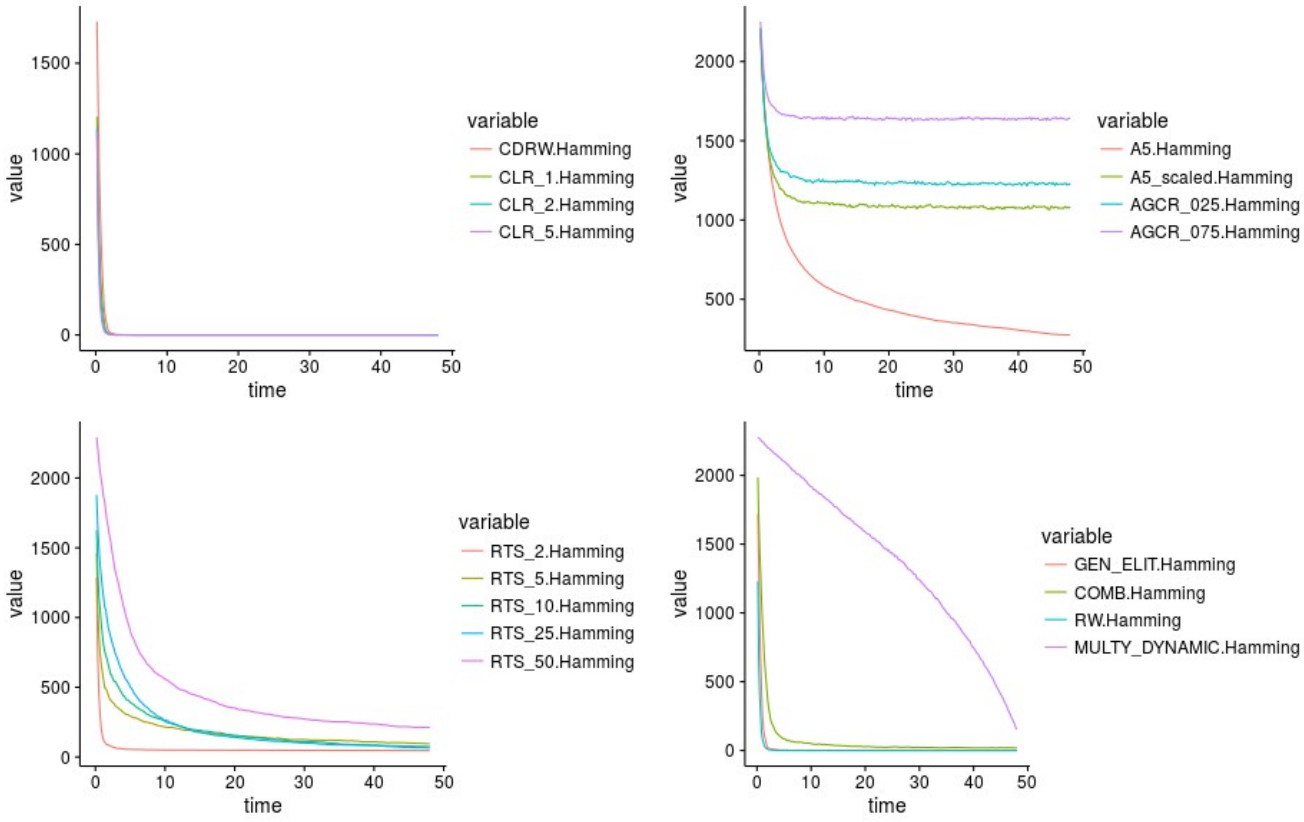
Media de la distancia al individuo mas cercano.

Para establecer esta métrica, utilizamos la *distancia de Hamming*, la cual consiste en el número de símbolos que son diferentes entre una cadena y otra. Para esta métrica se toma para cada población la distancia de Hamming entre cada individuo y los demás, tomando la menor distancia (el mas cercano); esto se hace para cada individuo de la población y se calcula el promedio de esas distancias, este valor nos da una idea de que tan diferentes son las solución que el algoritmo va generando en el transcurso del tiempo.



Gráfica 6. Evolución de la Entropía Poblacional para cada Algoritmo

En cuanto a la entropía poblacional, se observa claramente en la gráfica que la mayoría de los algoritmo pierden su diversidad casi inmediatamente, mientras que el MULTY_DYNAMIC conserva una medida de diversidad durante toda la ejecución, lo cual le permite generar siempre soluciones muy variadas y así obtener mejores resultados que los otros algoritmos que se estancan en un óptimo local.

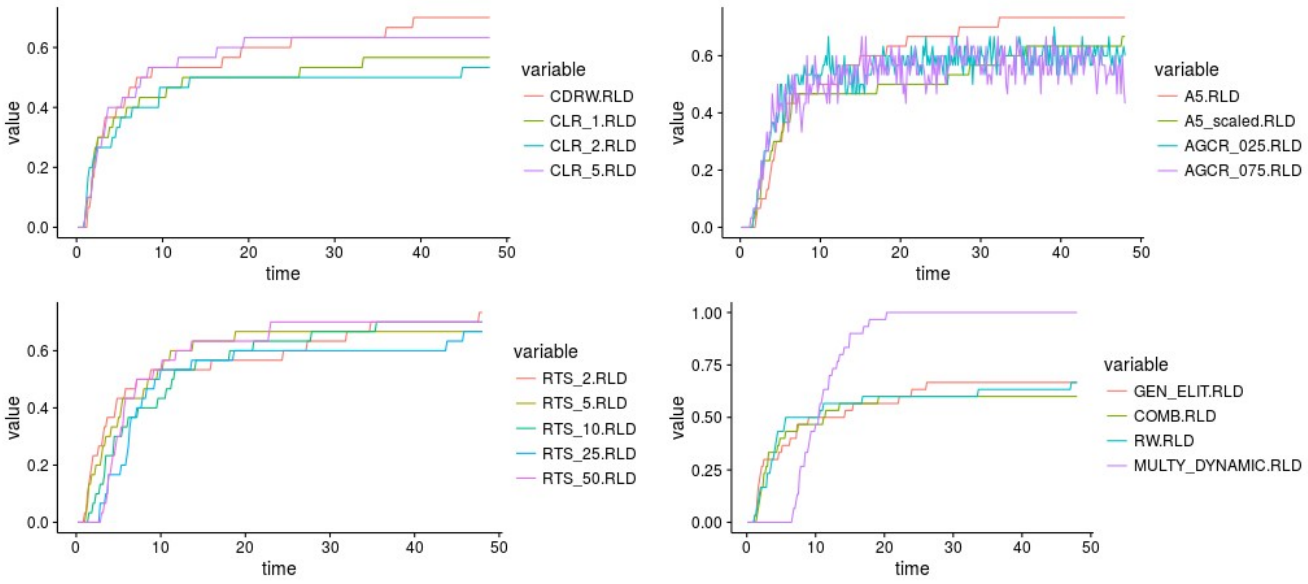


Gráfica 7. Media de la distancia al individuo mas cercano para cada Algoritmo

En el caso de la distancia Hamming al individuo mas cercano, se observa casi el mismo comportamiento, la mayoría pierde diversidad, en el caso de los algoritmos AGCR y A5_scaled, esta se conserva mas o menos constante, mientras que en el MULTY_DYNAMIC, se observa un descenso gradual.

vi) Run Length Distribution (RLD).

Se denominan también gráficas de convergencia, se simulan eligiendo un parámetro de calidad como referencia, que podría ser algún valor de media o mediana en cierto tiempo de la ejecución, para después evaluar el porcentaje de éxito de las ejecuciones que han alcanzado ese parámetro de calidad para cada tiempo de la ejecución.



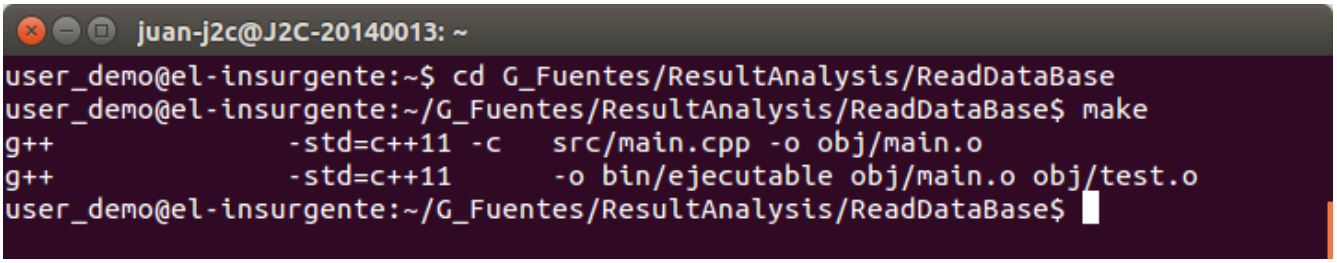
Gráfica 8. Run Length Dsitribution para cada Algoritmo, el parámetro de calidad elegido es la media a las 10 horas de ejecución.

Se observa en la gráfica que ningún algoritmo pudo converger al 100% de ejecuciones exitosas, con excepción del algoritmo MULTY_DYNAMIC, el cual desde las 20 horas ya había convergido en su totalidad.

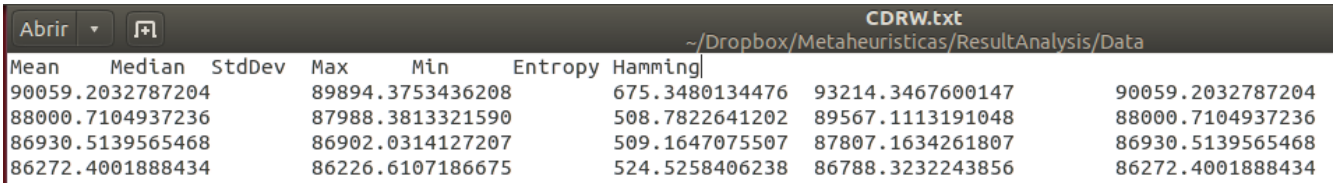
Compilación/Ejecución.

El programa que lee los archivos de ejecuciones se encuentra en la carpeta “ReadDataBase”, viene con un Makefile que soporta los comandos make, make run y make clean. Este programa recibe 4 argumentos:

1. Dirección del **archivo de ejecuciones** que se va a leer, sin la ultima cifra que va de 0 a 29, ya que el programa genera automáticamente esas extensiones para leer cada uno de los 30 archivos y construir la base de datos para realizar los cálculos.
2. Dirección del **archivo de salida**, en el cual el programa va a escribir por columnas los datos correspondientes a los métricos **para cada algoritmo** que vamos a utilizar durante esta practica para dibujar las gráficas, en total se generarán 17 archivos de salida.
3. Dirección del **archivo de resultados**, en este archivo se escribirá solamente el vector de los mejores valores de fitness para cada algoritmo, para después ser usado en los test estadísticos, solo habrá **un archivo para todos los algoritmos.**
4. Identificador del algoritmo (CDRW,AGCR,etc), mismo que se usará como encabezado al generar el archivo de resultados.

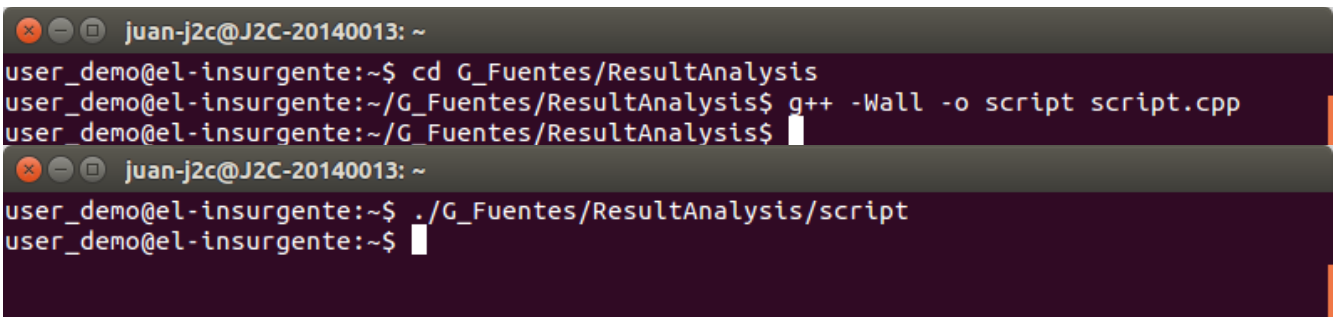


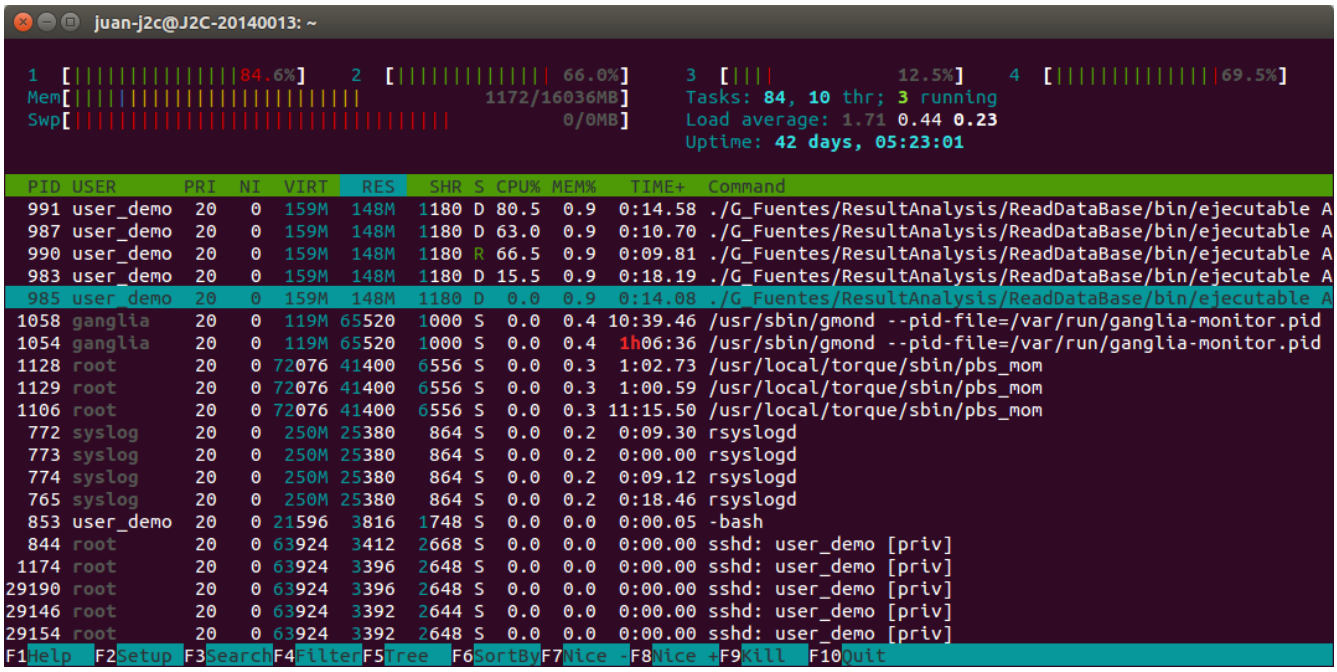
Compilación del programa que procesa la base de datos



Porción de un archivo de salida

Se incluye también el script para ejecutar el programa en el cluster, junto con el archivo de maquinas y el archivo de tareas, los 17 archivos de salida y el archivo de resultados se generan dentro de “Data” en la carpeta de “ResultAnalysis”.





Ejecución del procesador de la base de datos en el Insurgente

Se anexa también el programa en R donde se generaron todas las gráficas, el programa carga los 17 archivos y el archivo de resultados en data.frames para realizar los test estadísticos de Shapiro-Wilk, Kruskal-Wallis, Levene, Welch y Anova.