

Tarea 4

1985269

2 de abril de 2019

1. Descripción del experimento

Se escogieron tres generadores de grafos de la librería NetworkX, con cada uno de ellos se generaron cuatro grafos de distinto orden (logarítmico de base 3). De cada orden se obtuvieron diez grafos con densidades distintas. Luego se determinó el flujo máximo de cada grafo con cinco combinaciones de fuente sumidero distintas cinco veces más. Para calcular el flujo máximo se utilizaron tres algoritmos, para cada uno de ellos se realizó el procedimiento anterior.

Se almacenan los tiempos de ejecución de cada uno de los ciclos con el objetivo de estudiar los factores que lo afectan. Otras variables de interés estudiadas son: algoritmo, generador del grafo, cantidad de nodos y densidad.

Los algoritmos utilizados fueron:

- Maximum flow
- Edmonds Karp
- Boykov Kolmogorov

Generadores de Grafos:

- dense gnm random graph
- gnm random graph
- gnp random graph

El objetivo planteado es determinar si las variables de interés influyen en el tiempo de ejecución, para lo cual se realizó un análisis de varianzas y uno de correlación.

A continuación se comparte el código de Python con el que se recopiló la información:

```

1 Graf=nx.Graph()
2 rog=0
3 diccionario_inst_Shortest_path={}
4 while rog<=4:
5     Graf.clear()
6     rango=random.randint(random.randint(10,len(matrizadd)),len(matrizadd))
7     for i in range(rango):
8         for j in range(rango):
9             if matrizadd[i,j]!=0:
10                 matrizadd[j,i]=0
11                 #Graf.add_weighted_edges_from([(i,j,matrizadd[i,j])])
12                 Graf.add_edges_from([(i,j)])
13     cont=0
14     lista_tiempos=[]
15     lista_tiempos_completos={}
16     tiempo_de_paro=0
17     tiempo_inicial=0
18     tiempo_final =0
19     tiempo_ejecucion=0
20     for r in range(30):
21         lista_tiempos_completos[r+1]=[]
22         tiempo_de_paro=0
23         while tiempo_de_paro<1:
24             tiempo_inicial = time()
25             nx.shortest_path(Graf, source=None, target=None, weight=None,
26 method='dijkstra')
27             tiempo_final = time()
28             tiempo_ejecucion = tiempo_final - tiempo_inicial
29             if tiempo_ejecucion>0.0:
30                 lista_tiempos_completos[r+1].append((tiempo_ejecucion*10000))
31                 tiempo_de_paro+=tiempo_ejecucion
32         guardar_n_e[rog]=[]
33         diccionario_inst_Shortest_path[rog]=[]
34         for i in lista_tiempos_completos.keys():
35             media=np.mean(lista_tiempos_completos[i])
36             diccionario_inst_Shortest_path[rog].append(media)
37         guardar_n_e['nodos'].append(len(Graf.nodes))
38         guardar_n_e['edges'].append(len(Graf.edges))
39         guardar_n_e['media'].append(np.mean(diccionario_inst_Shortest_path[rog]))
40         guardar_n_e['dev'].append(np.std(diccionario_inst_Shortest_path[rog]))
41     rog+=1

```

2. Análisis de los datos

Como es posible observar en los diagrama caja bigote estos tienen una cola, es decir los tiempos se agrupan en un intervalo y luego se obtienen observaciones de tiempos mayores, lo cual puede tener efectos sobre la media ya que esta es muy sensible a puntos extremos. Lo que está sucediendo en terminos prácticos es que la mayoría de los tiempos se encuentran en cierto intervalo y cuando va aumentando el orden del grafo, el tiempo de ejecución va aumentando y salta bruscamente. Esto puede ser bien por la variación de las variables independientes combinado por el tiempo que demora el algoritmo de flujo máximo en resolver el problema.

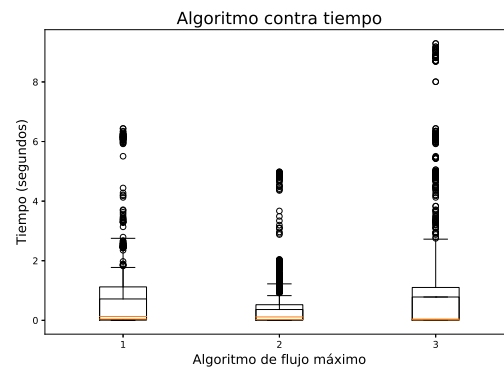


Figura 1: Algoritmo contra Tiempo

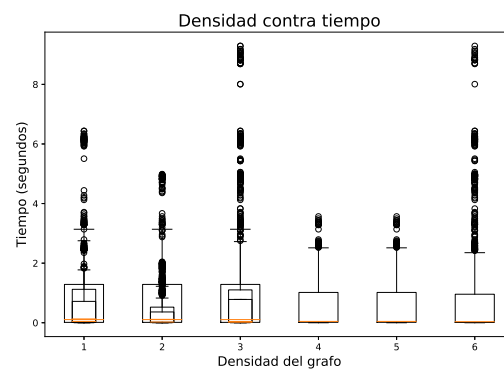


Figura 2: Densidad contra tiempo

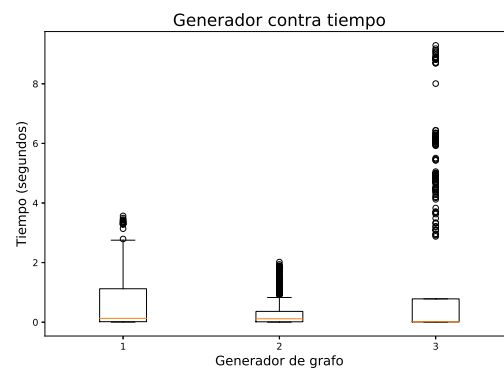


Figura 3: Generador contra grafo

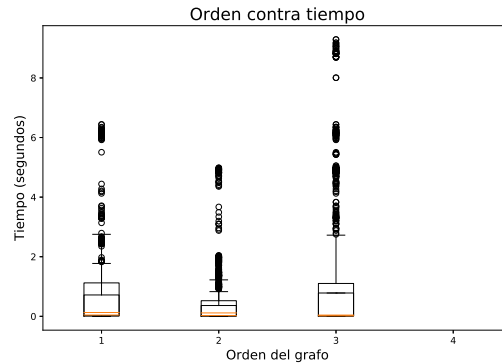


Figura 4: Orden contra tiempo

3. ANOVA

Se realizó un análisis de varianzas para cada uno de las variables con el objetivo de determinar si la medias con respecto al tiempo son diferentes. Como es posible observar en las salidas de la prueba el valor de P es muy pequeño, por lo que se puede afirmar que las medias de las variables con respecto al tiempo son diferentes. Entonces es posible concluir que las variables se relacionan con el tiempo de ejecución.

ANOVA.txt

	sum_sq	df	F	PR>F
Algoritmo	2.994151	1.0	4.623205	0.00
Generador	468.400575	1.0	723.247480	0.00
Orden	2999.136177	1.0	4630.903118	0.00
Densidad	733.071073	1.0	1131.919632	0.00
Residual	1162.505304	1795.0	NaN	NaN
c@FancyVerbLinee505304	1795.0	NaN	NaN	NaN

4. Mínimos cuadrados ordinarios

Para estudiar más a fondo la relación entre las variables y el tiempo se realizó una prueba de mínimos cuadrados ordinarios (OLS) por sus siglas en inglés. La salidas se muestran a continuación y el de R-squared indica que con estas variables es posible crear un modelo que elimine el setenta y cinco por ciento de los errores para determinar el tiempo. Del análisis de los P valores se puede reafirmar que la medias de las variables con respecto al tiempo son diferentes. La densidad es la más influyente según este análisis seguida por el generador y el orden, que aunque es menor se puede afirmar su relación con más confiabilidad que la de la variable algoritmo ya que esta tiene menor P valor. Además el análisis arroja que podría existir una fuerte multicolinealidad.

```

=====
                        OLS Regression Results
=====
Dep. Variable:          Tiempo      R-squared:                  0.757
Model:                  OLS         Adj. R-squared:            0.757
Method:                 Least Squares   F-statistic:              1399.
Date:                  Mon, 01 Apr 2019   Prob F-statistic:         0.00
Time:                  15:58:34         Log-Likelihood:           -2160.6
No. Observations:      1800            AIC:                     4331.
Df Residuals:          1795            BIC:                     4359.
Df Model:              4
Covariance Type:       nonrobust
=====

```

	coef	std err	t	P> t	[0.025	0.975]
Intercept	-3.0898	0.091	-34.136	0.000	-3.267	-2.912
Algoritmo	0.0500	0.023	2.150	0.032	0.004	0.096
Generador	0.6755	0.025	26.893	0.000	0.626	0.725
Orden	0.0047	6.89e-05	68.051	0.000	0.005	0.005
Densidad	1.9568	0.058	33.644	0.000	1.843	2.071

```

=====
Omnibus:                 762.497   Durbin-Watson:              0.441
ProbOmnibus:             0.000     Jarque-Bera JB:             6061.181
Skew:                    1.791     ProbJB:                     0.00
Kurtosis:                11.245     Cond. No.                   2.06e+03
=====

```

Warnings:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
 [2] The condition number is large, 2.06e+03. This might indicate that there are strong multicollinearity or other numerical problems.
 c@FancyVerbLineeinearity or other numerical problems.

5. Correlación y Comportamiento

En la figura 5 se muestra la matriz de correlación. Es posible observar que las correlaciones con el tiempo de las variables orden y densidad ratificando la información de la prueba anterior. Por último en la gráfica 2 se muestran en colores distintos cada uno de los algoritmos estudiados y puede observarse como a medida que aumenta el orden el tiempo crece abruptamente. Cada uno de los algoritmos fue representado con distintas formas.

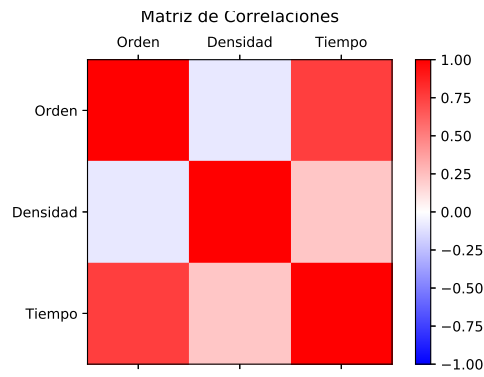


Figura 5: Correlación

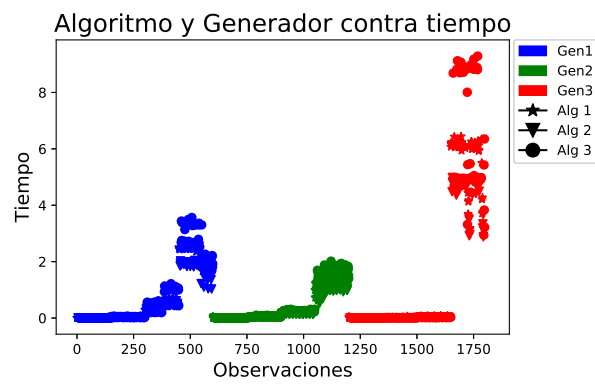


Figura 6: Comportamiento con respecto al tiempo

Referencias

- [1] Yehuda Koren. On spectral graph drawing. In *International Computing and Combinatorics Conference*, pages 496–508. Springer, 2003.
- [2] E. Schaeffer. <https://elisa.dyndns-web.com/>.
- [3] Python. <https://pythonfordatascience.org/anova-python/>.