

Caracterización estructural de instancias

Matrícula: 1985281

Resumen

Entre los generadores de grafos que proporciona [NetworkX](#), se seleccionó `random_geometric_graph` (grafo geométrico aleatorio) y la función `preflow_push` para calcular el flujo máximo entre un par de nodos, ya que resulta ser la más eficiente para este tipo de grafos. Además, se escogieron los grafos geométricos aleatorios, ya que se suelen utilizar para representar redes de sensores inalámbricos (inglés: wireless sensor networks (WSN)).

Introducción

Las redes de sensores inalámbricos se componen de nodos representados por los sensores, los cuales poseen una capacidad limitada de computación y comunicación. Por otro lado, la gran cantidad de sensores sobre una región dificulta la posibilidad de la colocación estratégica de nuevos dispositivos, y en consecuencia, la implementación aleatoria suele ser la mejor opción. Para este tipo de problemas se suele representar los sensores mediante puntos aleatorios finitos sobre la región, en donde las redes de sensores inalámbricos se modelan como grafos geométricos aleatorios que dependen del comportamiento probabilístico a estudiar [1].

Metodología y Resultados

Se visualizan cinco de las instancias producidas por el generador seleccionado y se visualizan con un acomodo que depende de las características del grafo, cambiando el tamaño y color de los nodos *fuente* (color verde) y *sumidero* (color rojo) en las visualizaciones, igual como el grosor de los arcos que son linealmente proporcionales a sus capacidades, ver figura 1.

```
1 G1 = nx.random_geometric_graph(50,0.25)
2 pos1 = nx.get_node_attributes(G1, 'pos')
3 weights1 = np.random.normal(3,1, nx.number_of_edges(G1))
4 w = 0
5 for u, v, d in G1.edges(data=True):
6     d['weight'] = weights1[w]
7     w += 1
8 f1 = {randint(0,49)}
9 s1 = {randint(0,49)}
10 nx.draw(G1, node_color='blue', edge_color='silver', node_size=80, width=weights1,
11         pos=pos1, with_labels=False, alpha= 0.7)
12 nx.draw_networkx_nodes(G1, pos1, nodelist=f1, node_size=150, node_color='green',
13                       node_shape='d')
14 nx.draw_networkx_nodes(G1, pos1, nodelist=s1, node_size=150, node_color='red',
15                       node_shape='d')
```

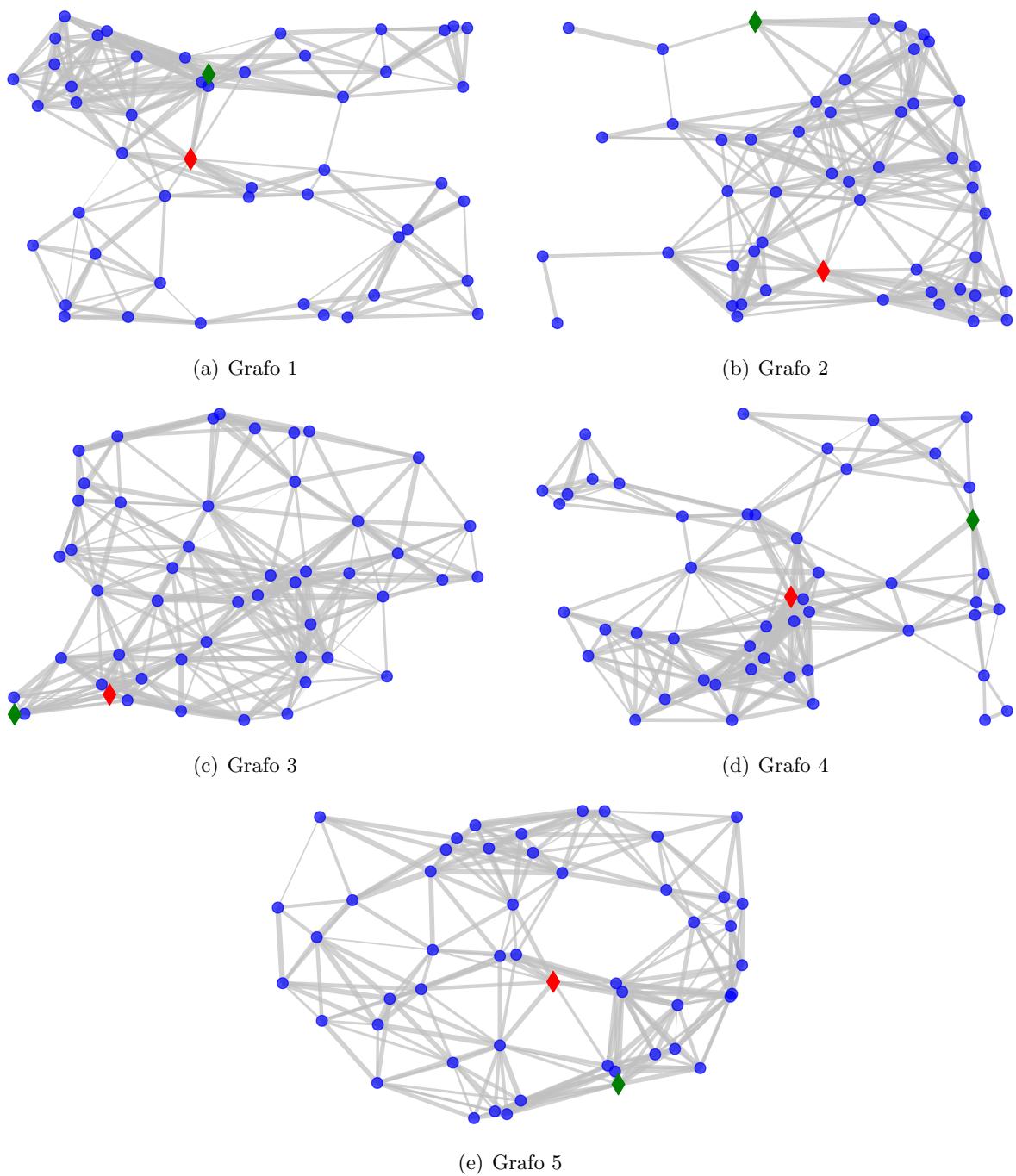


Figura 1: Ejemplo de cinco grafos geométricos aleatorios

Con los algoritmos disponibles en [NetworkX](#) [2], se calcula para las cinco instancias las siguientes características estructurales para todos sus vértices:

1. **Distribución de grado:** Calcula la distribución de probabilidad en los grados de los nodos en el grafo.
2. **Coeficiente de agrupamiento:** Calcula el número de triadas posibles para los nodos en el

grafo.

3. **Centralidad de cercanía:** Calcula la centralidad de cercanía de un nodo u como el recíproco de la suma de las distancias del camino más corto desde todos los $n - 1$ demás nodos.
4. **Centralidad de carga:** Calcula la centralidad de carga de un nodo u como la fracción de todas las rutas más cortas que pasan a través de ese nodo.
5. **Excentricidad:** Calcula la excentricidad de un nodo u como la distancia máxima de u a todos los demás nodos en el grafo.
6. **PageRank:** Calcula una clasificación de los nodos en el grafo en función de la estructura de los enlaces entrantes.

Cuadro 1: Instancias por características de sus nodos fuente y sumidero.

Grafo	Nodos		Característica estructural						Máximo Flujo
			1	2	3	4	5	6	
1	Fuente	49	5	0.40	0.28	0.024	6	0.022	2.48
	Sumidero	26	11	0.61	0.32	0.072	5	0.023	
2	Fuente	29	3	0.33	0.28	0.077	5	0.020	4.19
	Sumidero	12	5	0.90	0.29	0.003	6	0.018	
3	Fuente	22	8	0.67	0.37	0.050	5	0.018	5.31
	Sumidero	4	11	0.52	0.37	0.033	5	0.020	
4	Fuente	28	5	0.70	0.28	0.020	7	0.018	3.53
	Sumidero	2	5	0.60	0.35	0.079	5	0.015	
5	Fuente	5	3	0.70	0.28	0.026	6	0.018	3.06
	Sumidero	0	4	0.40	0.27	0.076	6	0.017	

Distribución de grado

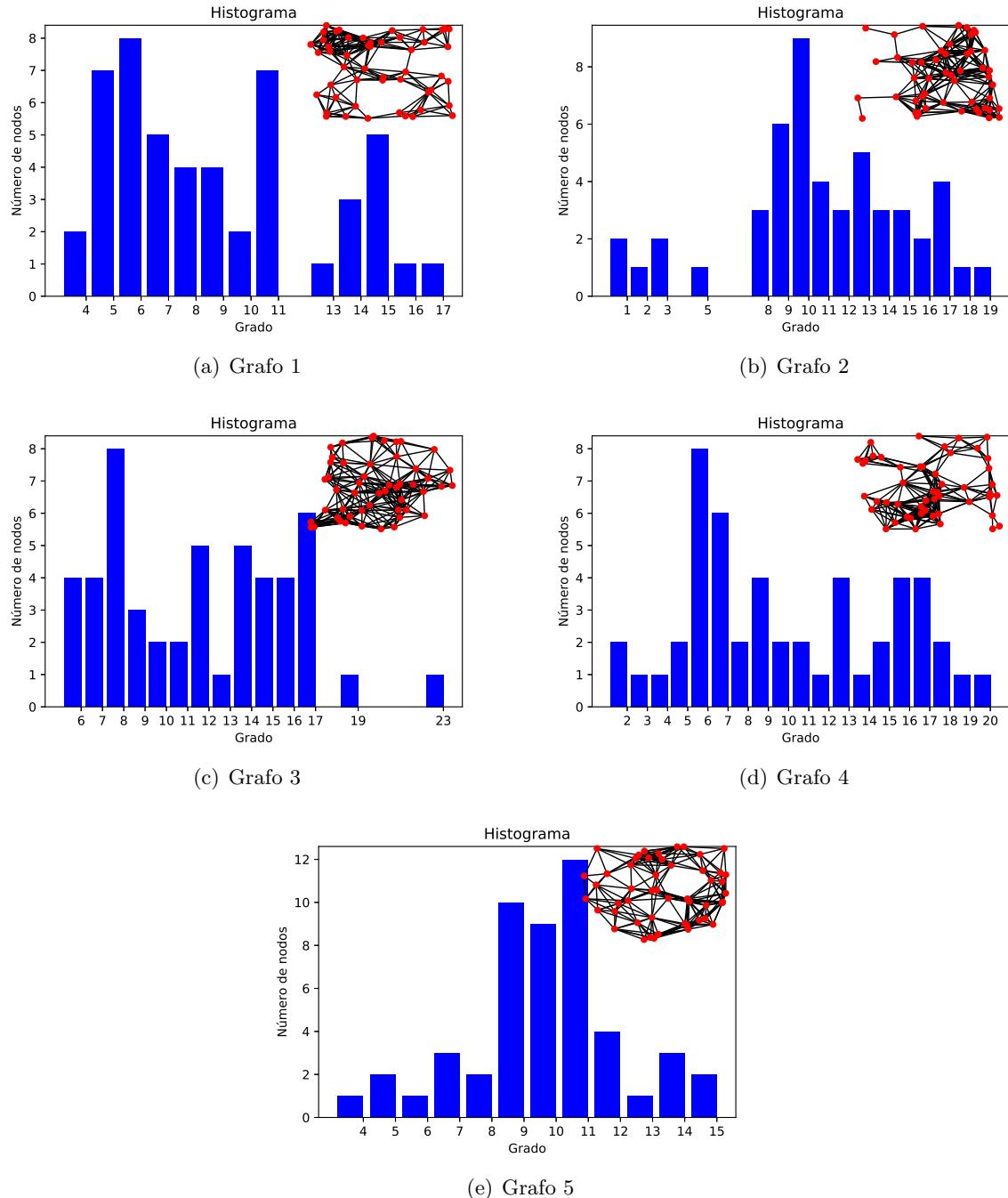


Figura 2: Histogramas de distribuciones de grado

Coeficiente de agrupamiento

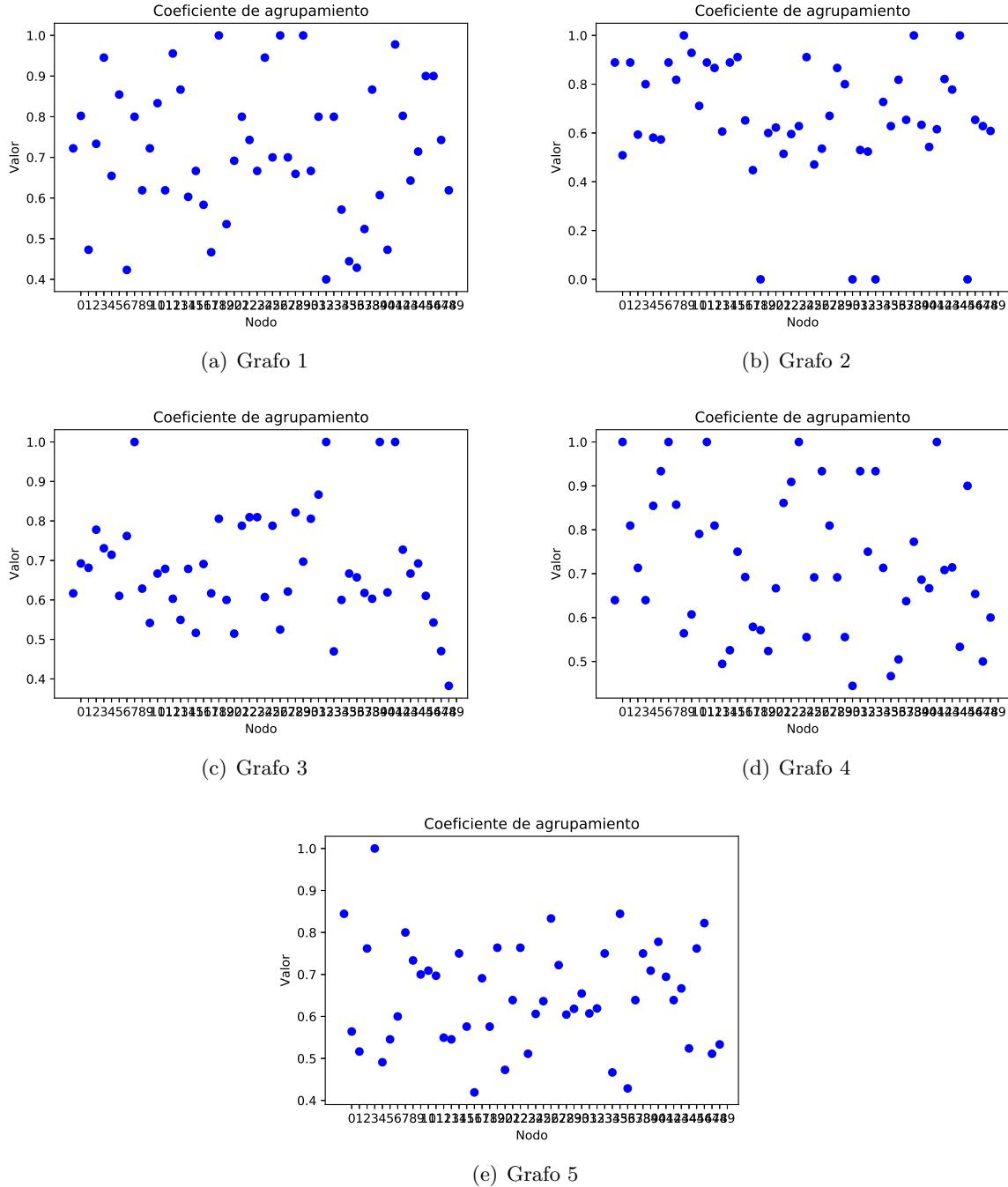
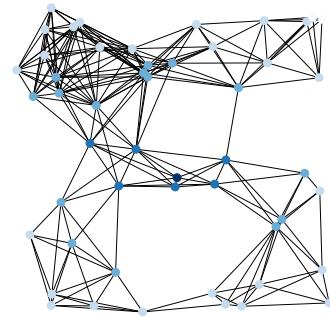


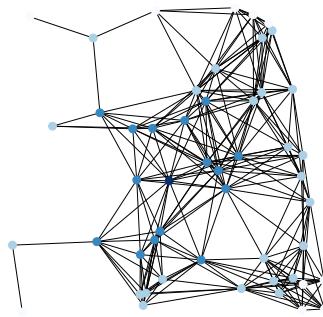
Figura 3: Coeficiente de agrupamiento

Centralidad de cercanía

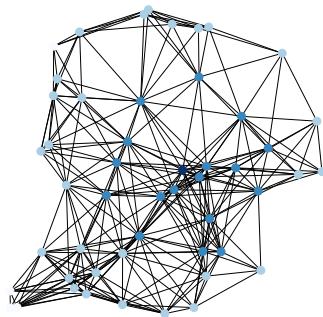
La figura 4 colorea los nodos dependiendo su valor de centralidad de cercanía, es decir, entre más oscuro es el nodo, quiere decir que tiene una centralidad de cercanía mayor, por lo que por ese nodo, suelen pasar más caminos cortos.



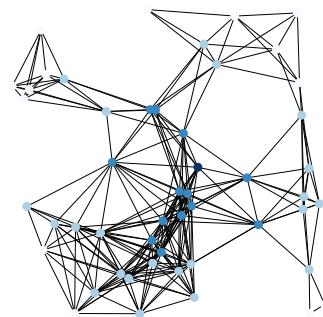
(a) Grafo 1



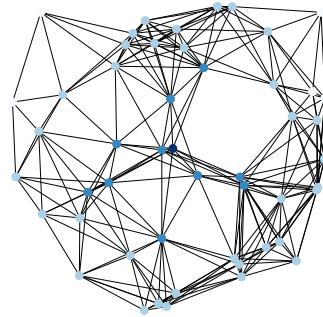
(b) Grafo 2



(c) Grafo 3



(d) Grafo 4



(e) Grafo 5

Figura 4: Centralidad de cercanía

Exentricidad

La figura 5 colorea los nodos dependiendo su valor de exentricidad, es decir, entre más oscuro es el nodo, quiere decir que tiene una exentricidad mayor, por lo que por ese nodo, pasan las

distancias máximas mas grandes.

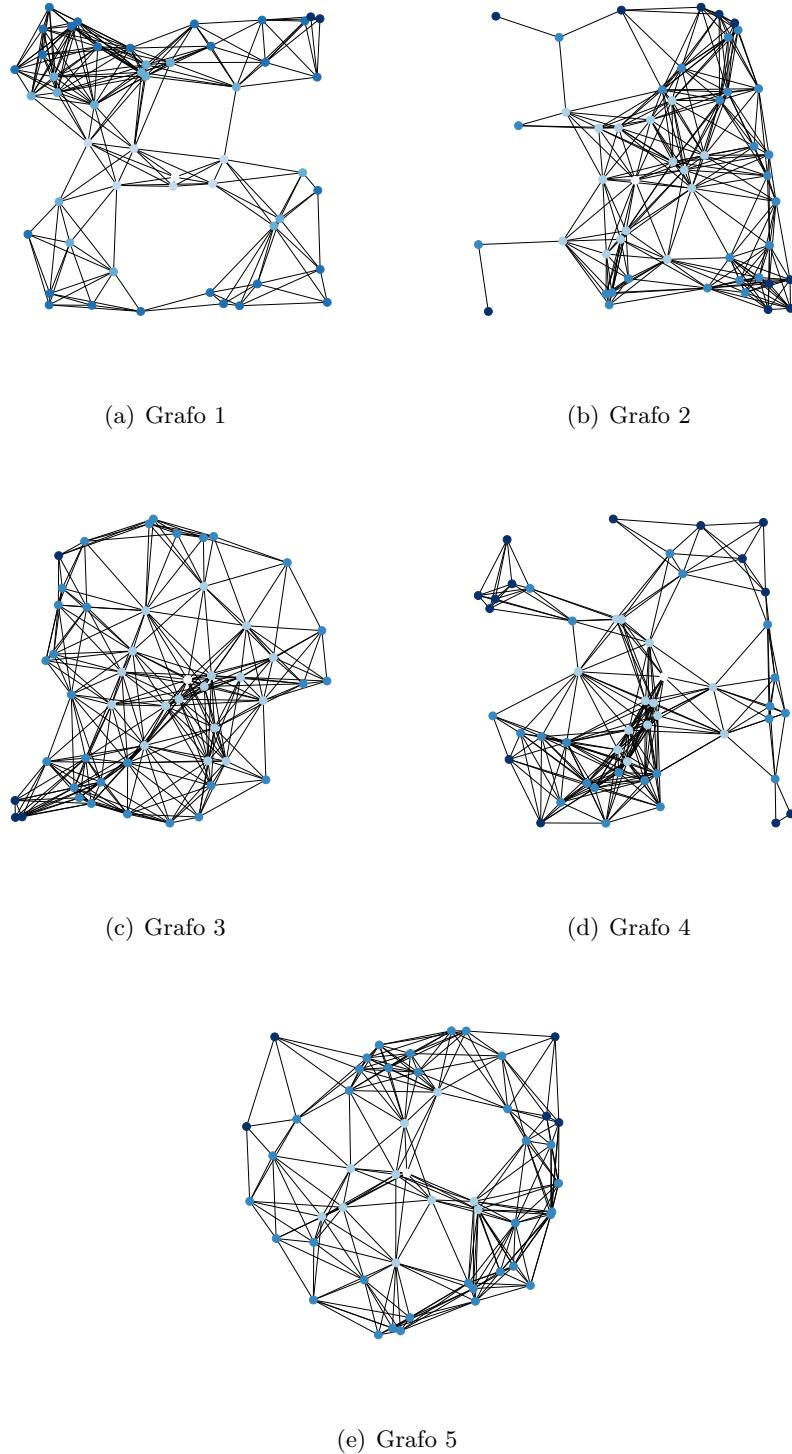
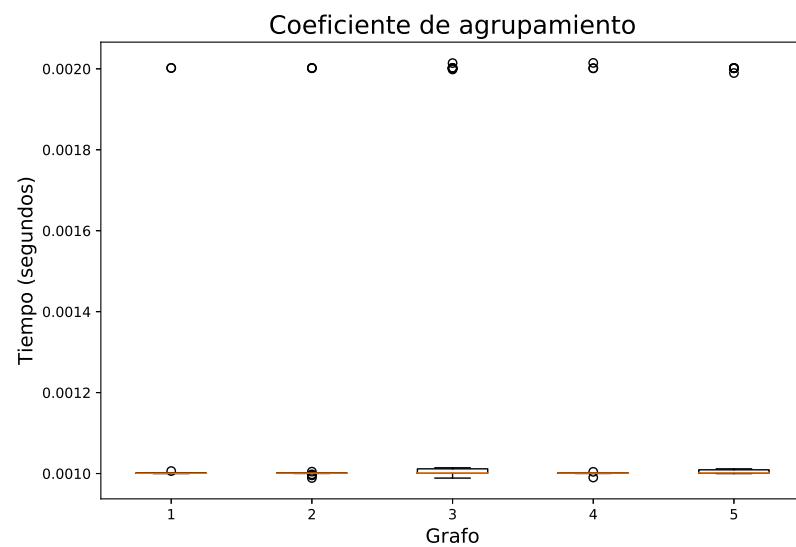
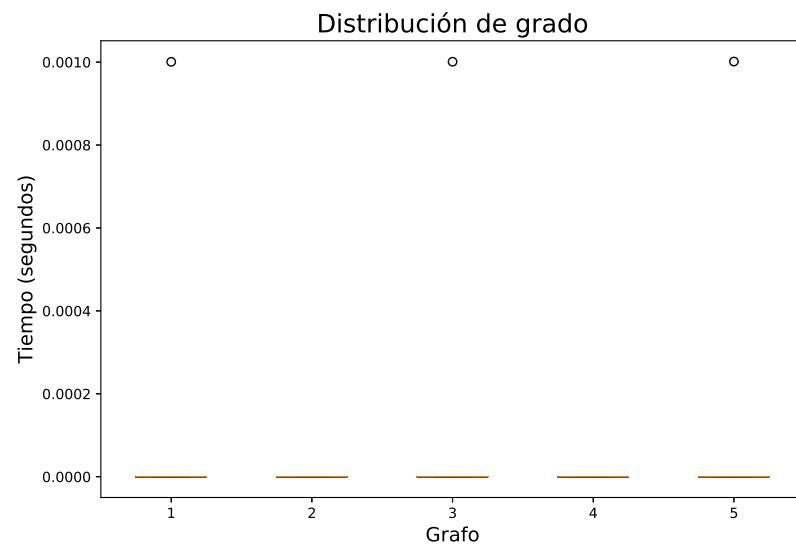
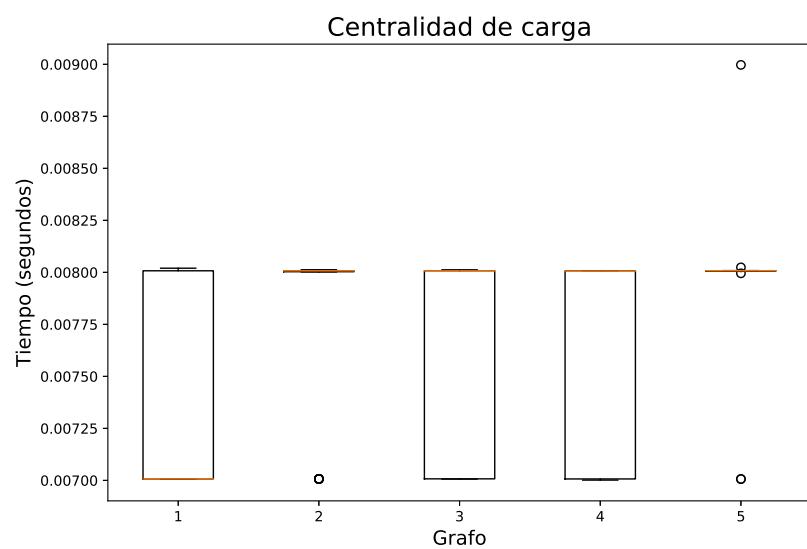
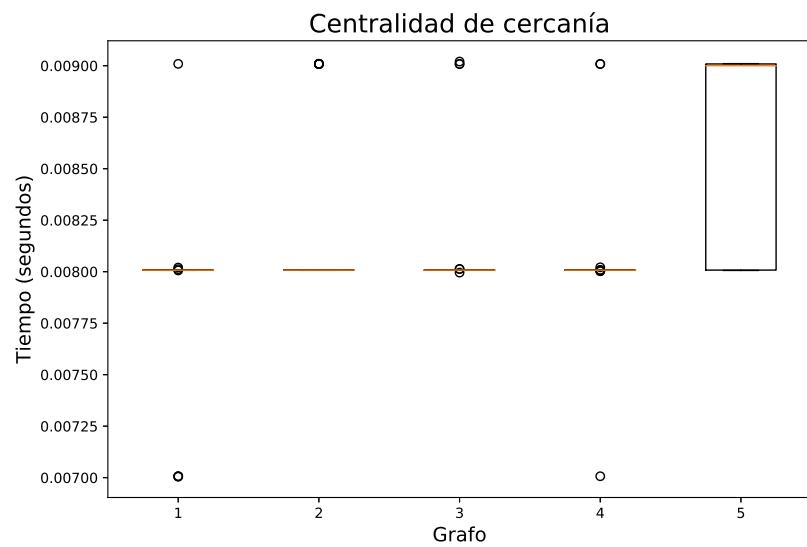
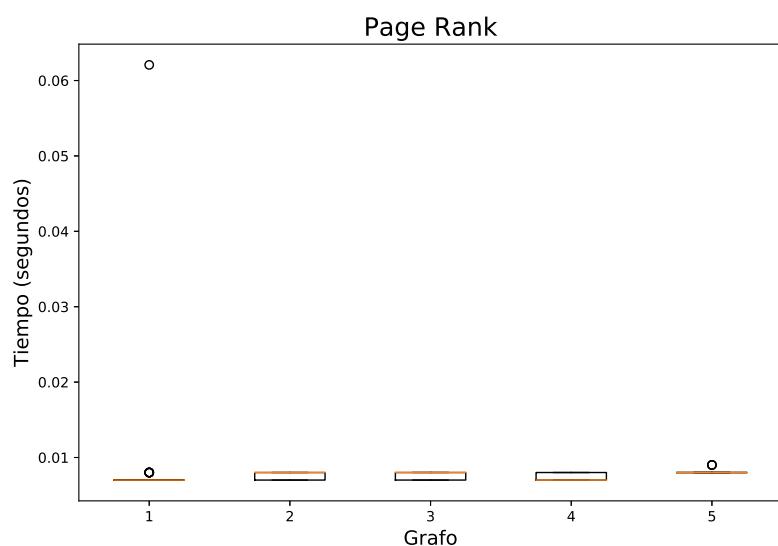
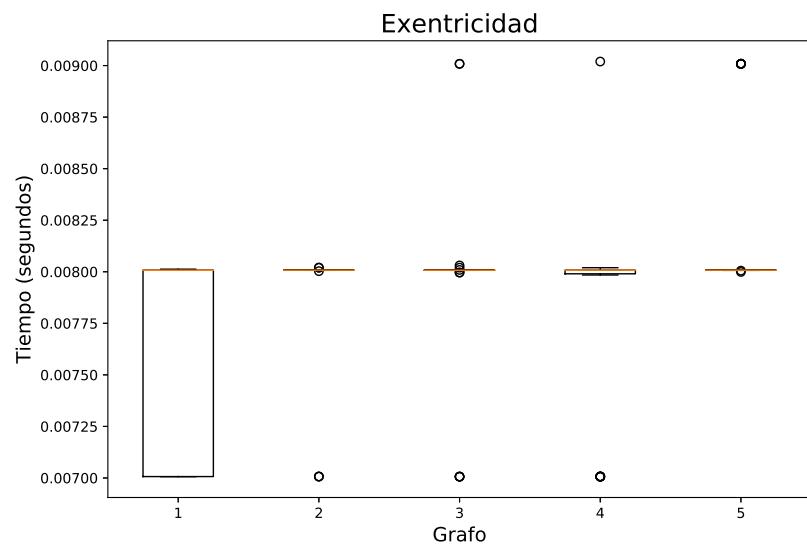


Figura 5: Exentricidad

A continuación se muestran los diagramas de caja y bigotes para los tiempos individuales de los cinco grafos utilizados al ejecutar las seis características estructurales.







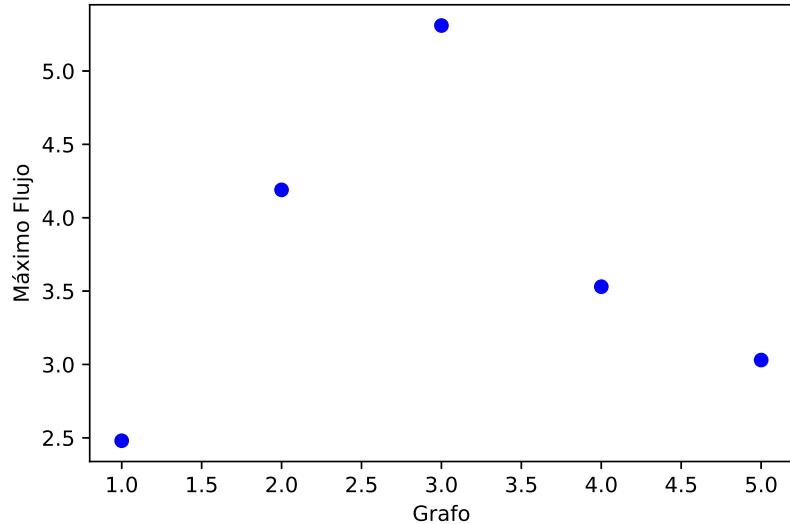


Figura 6: Grafos contra Valor óptimo

Conclusiones

De los diagramas de caja y bigote se puede apreciar estas características de los nodos no afectan al tiempo de ejecución del algoritmo seleccionado, sin embargo al ver la figura 6, se puede apreciar que al valor del óptimo si.

Si vemos las características en los nodos seleccionados como fuente y sumidero podemos ver que los nodos que cumplen las características del nodo fuente y nodo sumidero del grafo 3, resultan ser buenos nodos fuentes y buenos sumideros, ya que el valor del flujo máximo es mayor. Por otro, los nodos que cumplen las características del nodo fuente y nodo sumidero del grafo 1, sería mejor no usar como ninguno si uno busca obtener un alto flujo. Para ambos casos de nodos fuente y sumidero el tiempo de ejecución del algoritmo es rápido.

Referencias

- [1] Research Frontiers. Random geometric graphs and their applications, 2010. https://www.ugc.edu.hk/minisite/rgc_newsletter/rgcnews18/eng/05.htm. Accedido el 2019-04-29.
- [2] NetworkX. Algorithms. <https://networkx.github.io/documentation/networkx-2.2/reference/algorithms/index.html>. Accedido el 2019-04-29.
- [3] E. Schaeffer. Optimización de flujo en redes, 2019. <https://elisa.dyndns-web.com/teaching/opt/flow/>. Accedido el 2019-04-29.
- [4] A. Serna. Optimización de flujo en redes, 2019. <https://github.com/SernArmando>. Accedido el 2019-04-29.