

PRÁCTICA N° 3

Medidas de centralidad, comunidades y enlaces



- Módulo: Big Data Aplicado
- Unidad de trabajo: UND 1
- Nombre y apellidos: Alvaro Lucio-Villegas de Cea



Índice

Medidas de centralidad, comunidades y enlaces	1
Enunciado	3
Solución	4
Importación de CSVs	4
Ejemplos	5
Medidas de centralidad	5
Detección de comunidades	7
Predicción de enlaces	9



Enunciado

Realiza los ejemplos recogidos en el pdf a partir del punto 4.2.5. Medidas de centralidad, detección de comunidades y predicción de enlaces

Casi con toda probabilidad tendrás que realizar adaptaciones en el código debido a cambios en la librería



Solución

Importación de CSVs

```
1
2 WITH "https://github.com/neo4j-graph-analytics/book/raw/master/data/" + "social-nodes.csv" AS uri
3 LOAD CSV WITH HEADERS FROM uri AS row
4 MERGE (:User {id: row.id})
5
6 WITH "https://github.com/neo4j-graph-analytics/book/raw/master/data/" + "social-relationships.csv" AS uri
7 LOAD CSV WITH HEADERS FROM uri AS row
8 MATCH (source:User {id: row.src})
9 MATCH (destination:User {id: row.dst})
10 MERGE (source)-[:FOLLOWS]->([destination])
```

Added 9 labels, created 9 nodes, set 9 properties, created 16 relationships, completed after 5417 ms.

Ejemplos

Medidas de centralidad

- Centralidad de grado

```
CALL gds.degree.stream('CentralidadGrado')
```

```
YIELD nodeId, score
```

```
RETURN gds.util.asNode(nodeId).id AS name, score AS followers
```

```
ORDER BY followers DESC, name DESC
```

```
1 CALL gds.degree.stream('CentralidadGrado')
2
3
4 YIELD nodeId, score
5 RETURN gds.util.asNode(nodeId).id AS name, score AS followers
6 ORDER BY followers DESC, name DESC
```

name	followers
"Doug"	5.0
"Alice"	3.0
"Michael"	2.0
"Bridget"	2.0
"Mark"	2.0
"David"	1.0
"Charles"	1.0
"Amy"	1.0
"James"	0.0

MAX COLUMN WIDTH:

```
neo4j$ CALL gds.graph.project('CentralidadGrado','User',{FOLLOWS:{orientation:'REVERSE'}})
```

nodeProjection	relationshipProjection	graphName	nodeCount	relationshipCount	projectMillis
{ "User": { "label": "User", "properties": { } } }	{ "FOLLOWS": { "orientation": "REVERSE", "aggregation": "DEFAULT", "type": "FOLLOWS", "properties": { } } }	"CentralidadGrado"	9	16	7668

Started streaming 1 records after 26 ms and completed after 2263 ms.

- Cercanía

```
CALL
gds.beta.closeness.stream('Cercania',{nodeLabels:['User'],relationshipTypes:['FOLLOWS']})
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).id AS User, score
ORDER BY score DESC
```

```
1 CALL gds.beta.closeness.stream('Cercania',{nodeLabels:['User'],relationshipTypes:['FOLLOWS']})
2 YIELD nodeId, score
3 RETURN gds.util.asNode(nodeId).id AS User, score
4 ORDER BY score DESC
```

"User"	"score"
"David"	1.0
"Alice"	0.8833333333333334
"Bridget"	0.7142857142857143
"Michael"	0.7142857142857143
"James"	0.6666666666666666
"Mark"	0.625
"Doug"	0.4166666666666667
"Charles"	0.35714285714285715
"Amy"	0.0

MAX COLUMN WIDTH:

- Intermediación

```
CALL gds.betweenness.stream('Intermediacion')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).id AS name, score
ORDER BY score DESC
```

```
1 CALL gds.betweenness.stream('Intermediacion')
2 YIELD nodeId, score
3 RETURN gds.util.asNode(nodeId).id AS name, score
4 ORDER BY score DESC
```

"name"	"score"
"Alice"	10.0
"Doug"	7.0
"Mark"	7.0
"David"	1.0
"Bridget"	0.0
"Charles"	0.0
"Michael"	0.0
"Amy"	0.0
"James"	0.0

Detección de comunidades

- Conteo de triángulos

```
CALL gds.graph.project('CTriangulos2','User',{FOLLOWS: {orientation:
'UNDIRECTED'}})
```

```
CALL gds.triangleCount.stream('CTriangulos2')
YIELD nodeId, triangleCount
RETURN gds.util.asNode(nodeId).id AS name, triangleCount
ORDER BY triangleCount DESC
```

```
1 CALL gds.triangleCount.stream('CTriangulos2')
2 YIELD nodeId, triangleCount
3 RETURN gds.util.asNode(nodeId).id AS name, triangleCount
4 ORDER BY triangleCount DESC
```

"name"	"triangleCount"
"Alice"	5
"Doug"	5
"Bridget"	3
"Michael"	3
"Charles"	1
"Mark"	1
"David"	0
"Amy"	0
"James"	0

- Coeficiente local de clustering

```
CALL gds.localClusteringCoefficient.stream('CTriangulos2')
YIELD nodeId, localClusteringCoefficient
RETURN gds.util.asNode(nodeId).id AS name, localClusteringCoefficient
ORDER BY localClusteringCoefficient DESC
```

```
CALL gds.localClusteringCoefficient.stream('CTriangulos2')
YIELD nodeId, localClusteringCoefficient
RETURN gds.util.asNode(nodeId).id AS name, localClusteringCoefficient
ORDER BY localClusteringCoefficient DESC
```

"name"	"localClusteringCoefficient"
"Bridget"	1.0
"Charles"	1.0
"Mark"	1.0
"Michael"	1.0
"Alice"	0.5
"Doug"	0.5
"David"	0.0
"Amy"	0.0
"James"	0.0

- Componentes fuertemente conexas

```
CALL gds.alpha.scc.stream('CTriangulos2',{nodeLabels:['User'],
relationshipTypes:['FOLLOWS']})
YIELD nodeId, componentId
RETURN gds.util.asNode(nodeId).id AS Name, componentId AS Component
ORDER BY Component DESC
```

```
1 CALL gds.alpha.scc.stream('CTriangulos2',{nodeLabels:['User'], relationshipTypes:['FOLLOWS']})
2 YIELD nodeId, componentId
3 RETURN gds.util.asNode(nodeId).id AS Name, componentId AS Component
4 ORDER BY Component DESC
5
```

Name	Component
David	6
Amy	6
James	6
Alice	0
Bridget	0
Charles	0
Doug	0
Mark	0
Michael	0

Predicción de enlaces

- Vecinos comunes

```
MATCH (x:User{id:'Charles'})
MATCH (y:User{id:'Bridget'})
RETURN gds.alpha.linkprediction.commonNeighbors(x,y) AS score
```

```
1 MATCH (x:User{id:'Charles'})
2 MATCH (y:User{id:'Bridget'})
3 RETURN gds.alpha.linkprediction.commonNeighbors(x,y) AS score
4
```

score
2.0

- Adhesión preferencial

```
MATCH (x:User{id:'Charles'})
MATCH (y:User{id:'Bridget'})
RETURN gds.alpha.linkprediction.commonNeighbors(x,y) AS score
```

```
1 MATCH (x:User{id:'Charles'})
2 MATCH (y:User{id:'Bridget'})
3 RETURN gds.alpha.linkprediction.commonNeighbors(x,y) AS score
4
```

Table	"score"
Text	2.0

- Asignación de recursos

```
MATCH (x:User{id:'Charles'})
MATCH (y:User{id:'Bridget'})
RETURN gds.alpha.linkprediction.resourceAllocation(x, y) AS score
```

```
MATCH (x:User{id:'Charles'})
MATCH (y:User{id:'Bridget'})
RETURN gds.alpha.linkprediction.resourceAllocation(x, y) AS score
```

"score"
0.30952380952380953