

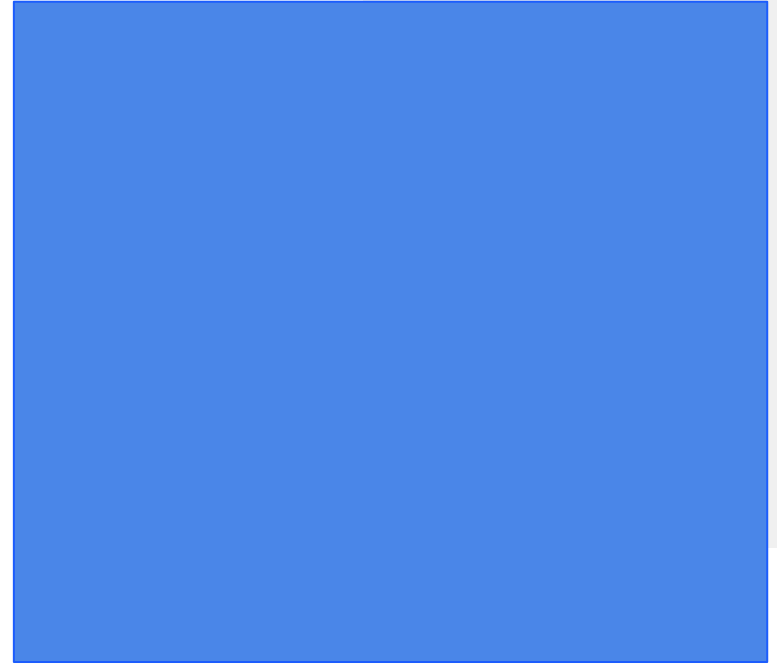


Big Data Analytics

Dr Sirintra Vaiwsri | Email: sirintra.v@itm.kmutnb.ac.th



Graph Analytics



Graph Analytics (Chambers and Zaharia, 2018)


- Graphs are data structures that contain nodes which are objects and edges which define the relationships between these objects (nodes).
- Graph analytics is the process to analyse these relationships.
- Nodes and edges in a group can have data associated with them.
- For example, a graph can be used to represent a friend group, where each node is a person and each edge is the weight of interactions between friends.
 - Low weight might represent rare interaction while high weight might represent frequent interaction between friends.

GraphFrames (Chambers and Zaharia, 2018)

- To work with GraphFrame, you first need to install the GraphFrame package
- To create a GraphFrame, nodes and edges Dataframes are required.
- The nodes DataFrame must contain a column named “id”.
- The edges DataFrame must contain columns “src” and “dst” which are the source nodes and destination nodes, respectively.



Querying the Graph (Chambers and Zaharia, 2018)

- You can group and order the nodes and edges in the GraphFrame
 - `<your graph>.edges.groupBy("src",
"dst").count().orderBy(desc("count")).show(5)`
 - You can also filter the GraphFrame
 - `<your graph>.edges.where("src = '<something>' OR dst
= '<something>').groupBy("src",
"dst").count().orderBy(desc("count")).show(5)`
- 

Subgraph (Chambers and Zaharia, 2018)

- Subgraphs are smaller graphs within the graph.
- You can use the query to create a subgraph.
 - `<your query name> = <your graph>.edges.where("src = '<something>' OR dst = '<something>'")`
 - `<your subgraph> = GraphFrame(<your graph>.vertices, <your query name>)`

Motif Finding (Chambers and Zaharia, 2018)

- Motifs are a way to query for patterns in the data instead of actual data.
- The query is specified in a domain-specific language.
- This language lets you specify combinations of vertices and edges.
- It is also allowed for assigning the names.
- For example, nodes a and b are connected through an edge ab :
 - $(a) - [ab] \rightarrow (b)$
 - These names a , b , and ab are the names of columns in the linked results in DataFrame.

Motif Finding (Chambers and Zaharia, 2018)

- You can find motifs using “find()”.
- For example:
 - motifs = <your graph>.find(“(a) - [ab] -> (b)”)
 - motifs = <your graph>.find(“(a) - [ab] -> (b)”).filter(“ab.<your column> = ‘<some value>’ OR ab.<your column> = ‘<some value>’”)


PageRank (Chambers and Zaharia, 2018)

- PageRank works by counting the number and quality of edges to a node to estimate the importance of the node.
- The assumption is that if the node is important, it is more likely that the node will be linked (via edges) from other nodes.
- Example:
 - `<your rank> = <your graph>.pageRank(resetProbability = 0.15, maxIter = 5)`
 - `<your rank>.vertices.orderBy(desc("pagerank")).show(5)`
- `resetProbability` is the probability of resetting to a random node (vertex)



In-Degree and Out-Degree

(Chambers and Zaharia, 2018)

- Directed Graph has start and end nodes.
 - Thus, each node has a link (edge) into or out of that node.
 - To measure the in and out of the node, you can use the in-degree and out-degree metrics.
 - Example:
 - `<your in-degree> = <your graph>.inDegrees`
 - `<your in-degree>.orderBy(desc("inDegree")).show(5)`
- 

Connected Components

(Chambers and Zaharia, 2018)

- A connected component is an undirected subgraph that only has a connection to itself but does not connect to the greater graph.
- A strongly connected component is a directed subgraph that has links between all pairs of nodes inside it.
- To use these connected components, the checkpointDir is required
 - `spark.sparkContext.setCheckpointDir("/tmp/checkpoints")`
- `<your cc> = <your graph>.connectedComponents()`
- `<your scc> = <your graph>.stronglyConnectedComponents(maxIter = 5)`

Breadth-First Search (Chambers and Zaharia, 2018)

- You can search your graph using the Bread-First Search (BFS) algorithm.
- You can specify the maximum of edges to follow using “maxPathLength”.
- Example,
 - `<your graph>.bfs(fromExpr = “id = ‘<some id>’”),
toExpr = “id = ‘<some id>’”, maxPathLength =
2).show()`



Graph Analytics Implementation



- Import Libraries:
 - SparkSession
 - GraphFrame from graphframes
 - desc, col from pyspark.sql.functions

Graph Analytics Implementation

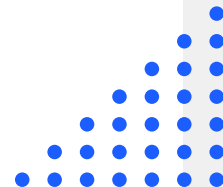
Use vertices and edges as shown in the figure:

```
# Create vertices (nodes) dataframe
vertices = spark.createDataFrame([
    ("Alice", 45),
    ("Jacob", 43),
    ("Roy", 21),
    ("Ryan", 49),
    ("Emily", 24),
    ("Sheldon", 52)],
    ["id", "age"])

# Create edges dataframe
edges = spark.createDataFrame([("Sheldon", "Alice", "Sister"),
    ("Alice", "Jacob", "Husband"),
    ("Emily", "Jacob", "Father"),
    ("Ryan", "Alice", "Friend"),
    ("Alice", "Emily", "Daughter"),
    ("Alice", "Roy", "Son"),
    ("Jacob", "Roy", "Son")],
    ["src", "dst", "relation"])
```

Assignment (2 points)

- Please implement the graph analytics following slides 5-12 and show the results to get 2 points.





References

- Chambers, B., & Zaharia, M. (2018). Spark: The definitive guide: Big data processing made simple. "O'Reilly Media, Inc."