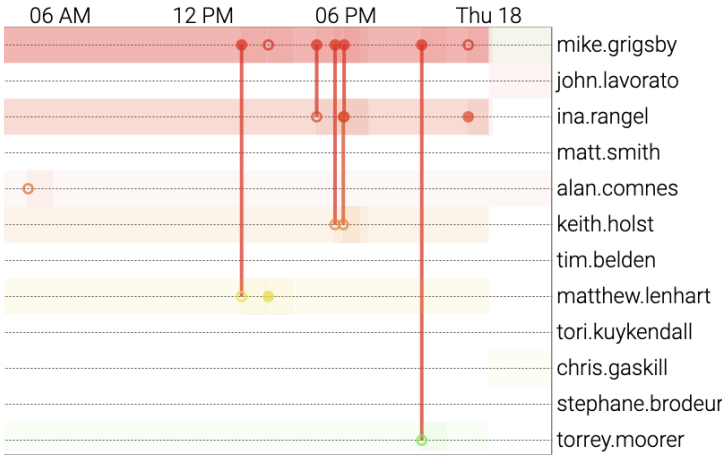
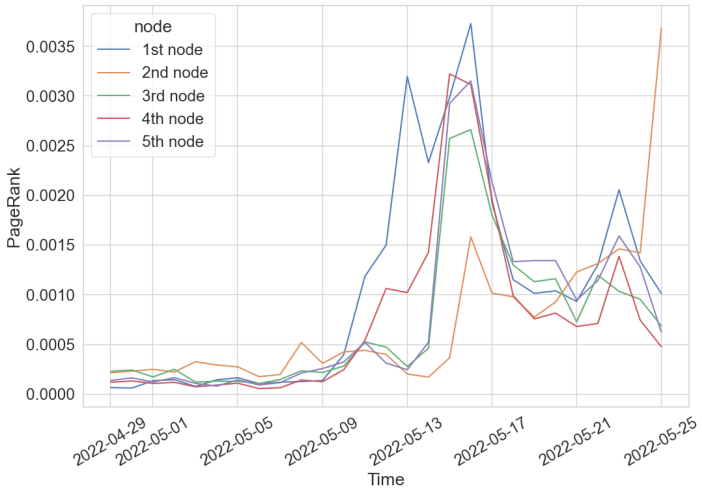
```
import raphtory as rp
import pandas as pd

enron = pd.read_csv("enron.csv", names = ["time",
  "src",
    "dst",
      "message",
        "type"])

g = rp.Graph()

for id, record in enron.iterrows():
      g.add_edge(timestamp = record['time'],
        src = record['src'],
         dst = record['dst'],
           properties = {"message": record['message']},
             layer = record['type'])
```




```
from raphtory.plottingutils import ordinal_number

# Load graph of transactions
g = rp.graph_loader.stable_coin_graph(path = dataset_path,
  subset=True)
layer_graph = g.layer("LUNC")

# Get top 5 nodes by PageRank
pg = rp.algorithms.pagerank(g = layer_graph)
top_5, scores = zip(*pg.top_k(5))

# Extract their daily PageRank value
metric_list = []
graph_views = layer_graph.rolling(window = '1 day',
  step = '1 day')

for view_g in graph_views:
    time = view_g.end_date_time()
    values = pagerank(g = view_g)
    for i, n in enumerate(top_5_nodes):
        metric_list.append( (time,
          values[n],
            f"{ordinal_number(i+1)} node") )

# Plot these values
plotting_function(metric_list, x = "Time",
  y = "PageRank", hue = "node")
```




```
from raphtory.algorithms import global_temporal_three_node_motifs
from raphtory.plottingutils import global_motif_heatplot

# Load the graph using the load-from-pandas functionality
df = pd.read_csv("sx-mathoverflow.txt", names=["src", dst",
  "time"])
g = rp.Graph()
g.load_edges_from_pandas(df,src="src",dst="dst",time="time")

# Count the 1-hour motifs for g
counts = global_temporal_three_node_motifs(g, delta=3600)

# Out-of-the-box motif plot
ax = global_motif_heatplot(counts, cmap='YlGnBu')
```