Analysis

November 29, 2019

[260]:
```
import os
import pandas as pd
import matplotlib.pyplot as plt

from datetime import datetime
```

1 Utilities and analysis setup

[261]:
```
# dict to map node ID to node address
node_map = {
    '1': 'planetlab2.citadel.edu',
    '2': 'planetlab2.c3sl.ufpr.br',
    '3': 'planetlab6.goto.info.waseda.ac.jp',
    '4': 'pl-dccd-01.cua.uam.mx',
    '5': 'planetlab3.rutgers.edu',
    '6': 'planetlab2.ie.cuhk.edu.hk',
    '7': 'planetlab1.temple.edu',
    '8': 'planetlab1.rutgers.edu'
}

node_loc = {
    '1': 'US (Citadel)',
    '2': 'Brazil',
    '3': 'Japan',
    '4': 'Mexico',
    '5': 'US (Rutgers 3)',
    '6': 'Hong Kong',
    '7': 'US (Temple)',
    '8': 'US (Rutgers 1)'
}

node_pairs = {
    '1': '1-2 & 2-1'
}
```

```
# dict to map each node's to its respective timezone
```

1.0.1 Set the current working directory to the project root

[262]:
```
os.chdir("/Users/jlahut/UAlbany/comp-comm-networks/final-project/project/")
```

1.0.2 Read in data files

Data files have the following format <type>_<src>-<dest>_<Y-M-D>_<H-M-S>
The timestamps in the file names are local to the node in which it came from

[263]:
```
data = []
for filename in os.listdir("data/"):
    try:
        items = filename.split('_')

        measure = items[0]
        src, dest = items[1].split('-')

        date = items[2]
        time = items[3].split('.')[0]

        timestamp = datetime.strptime(f'{date} {time}', '%Y-%m-%d %H-%M-%S')
        content = ''

        with open(os.path.join('data', filename), 'r') as file:
            content = file.read()

        if (src == '1' and dest == '2') or (src == '2' and dest == '1'):
            pair = '1-2 & 2-1'
        elif (src == '3' and dest == '5') or (src == '5' and dest == '3'):
            pair = '3-5 & 5-3'
        elif (src == '4' and dest == '7') or (src == '7' and dest == '4'):
            pair = '4-7 & 7-4'
        elif (src == '6' and dest == '8') or (src == '8' and dest == '6'):
            pair = '6-8 & 8-6'
        elif (src == '1' and dest == '5') or (src == '5' and dest == '1'):
            pair = '1-5 & 5-1'
        else:
            pair = 'Error'

        data.append([src, dest, timestamp, measure, node_map[src],␣
↪node_map[dest], content, filename, pair])
    except:
        print(f'Could not read file: {filename}')
```

```
Could not read file: traceroute_-_2019-11-21_01-36-28.txt
```

1.0.3 Create 'master' dataframe to hold all raw data

[274]:
```
df = pd.DataFrame(data, columns = ['src_id', 'dest_id', 'time', 'measure',␣
↪'src_name', 'dest_name', 'raw_data', 'filename', 'pair'])
```

1.0.4 Ping and Traceroute functions to be applied accross the dataframe

- Used mainly for file parsing

[275]:
```
# example format
# rtt min/avg/max/mdev = 68.244/68.337/68.498/0.129 ms or
# rtt min/avg/max/mdev = 68.244/68.337/68.498/0.129 ms, pipe n
# 20 packets transmitted, 20 received, 0% packet loss, time 19029ms
def analyze_ping(data):
    try:
        # --- parse out statistics ---
        # second to last line will always be calculated values
        # split on '=' sign, then strip all spaces, remove units, and finally␣
↪split on '/'
        # giving the values we need
        metrics = data['raw_data'].split('\n')[-2].split('=')[1].strip().
↪replace(' ms', '').split(',')[0].split('/')
        data['min_ping_time'], data['avg_ping_time'], data['max_ping_time'],␣
↪data['sd_ping_time'] = \
            float(metrics[0]), float(metrics[1]), float(metrics[2]),␣
↪float(metrics[3])

        # --- parse out packet loss ---
        # third to last line will always be the line with packet loss
        data['packet_loss'] = float(data['raw_data'].split('\n')[-3].
↪split(',')[2].split(' ')[1].replace('%', ''))
    except Exception as e:
        print(f"No calculated data for: {data['filename']}")
    return data

def analyze_traceroute(data):
    # TODO
    return data
```

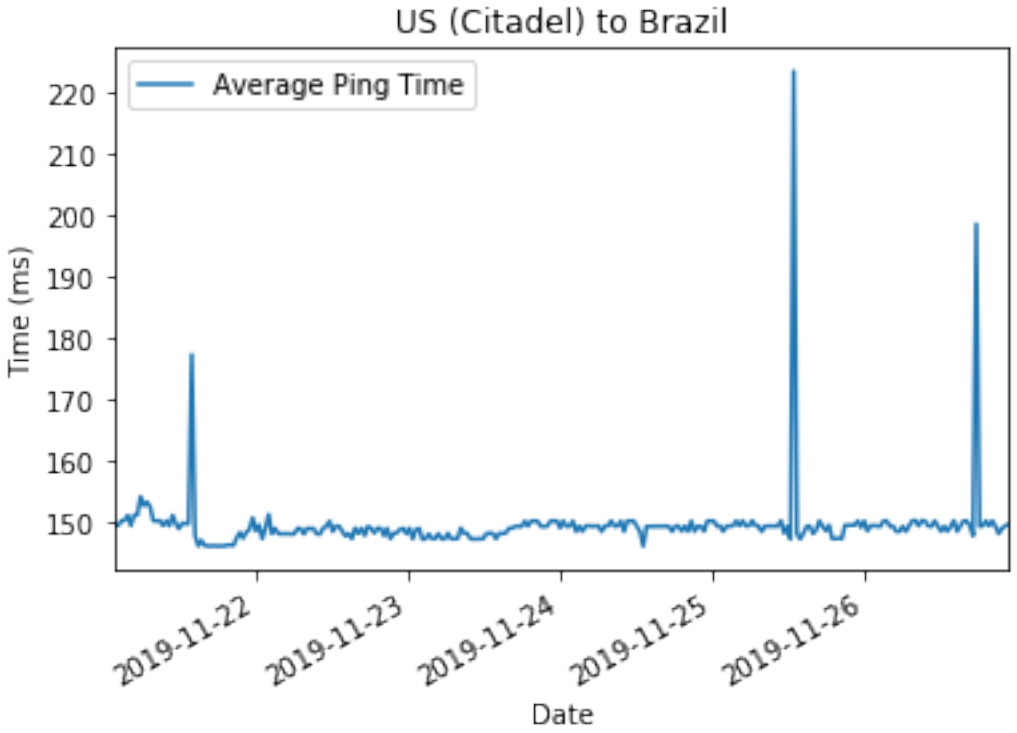
1.0.5 Apply the functions

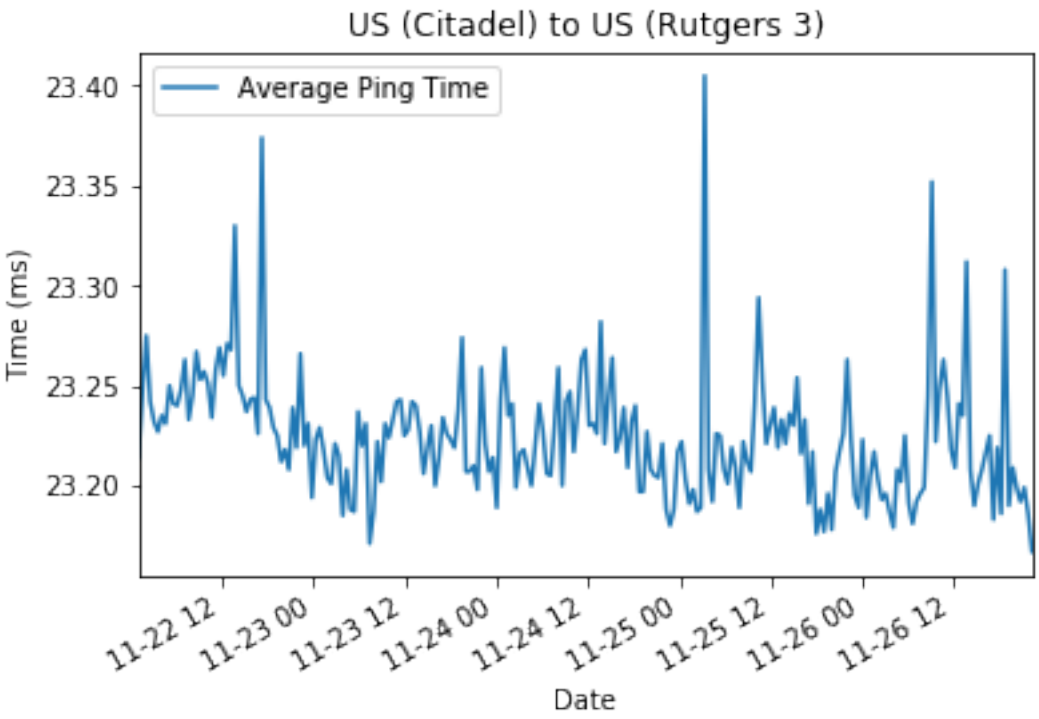
[276]:
```
df = df[df['measure'] == 'ping'].apply(analyze_ping, axis = 1)
# df = df[df['measure'] == 'traceroute'].apply(analyze_traceroute, axis = 1)
```

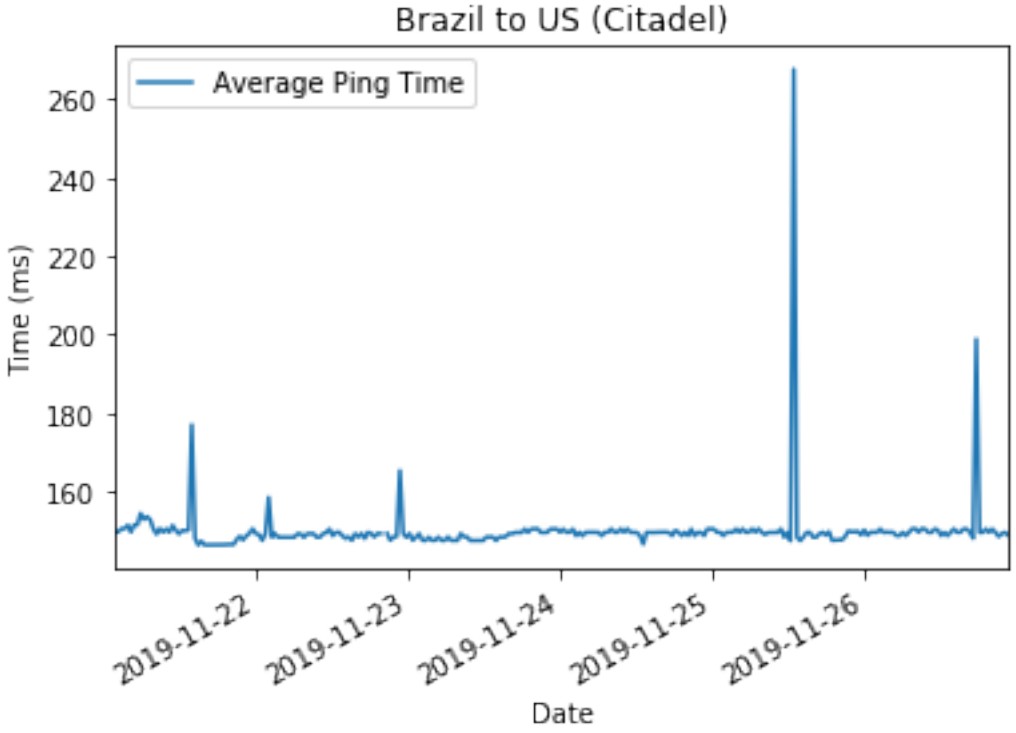
```
No calculated data for: ping_8-6_2019-11-25_18-34-59.txt
No calculated data for: ping_5-3_2019-11-25_18-09-07.txt
No calculated data for: ping_2-1_2019-11-22_20-02-21.txt
```

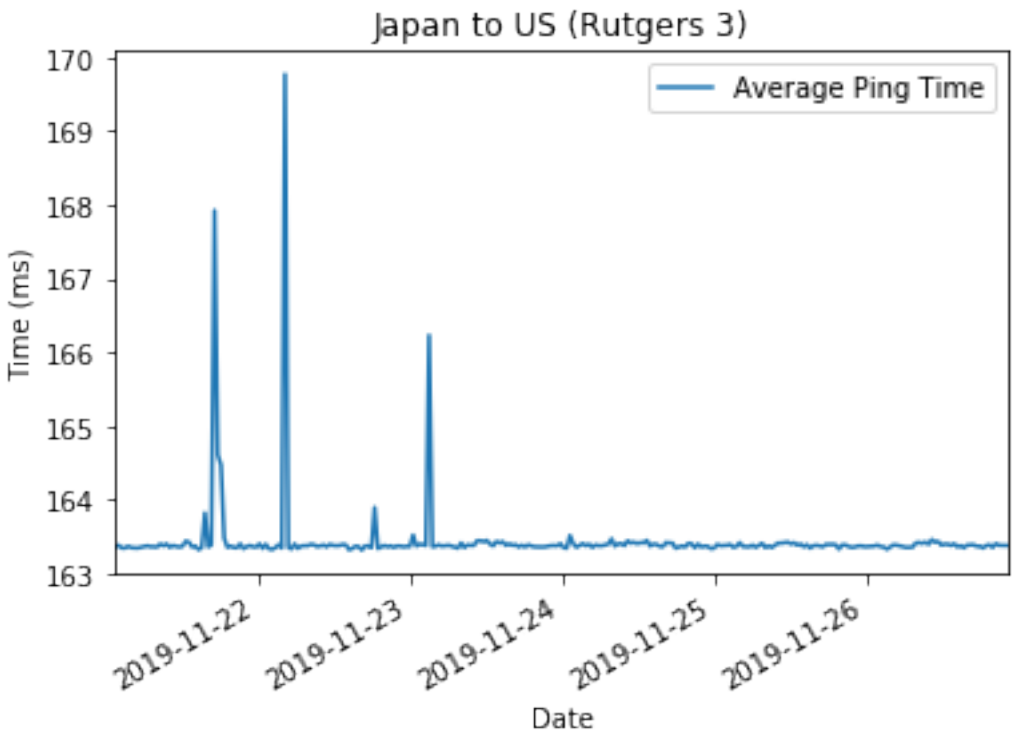
1.0.6 Plotting average ping times for nodes

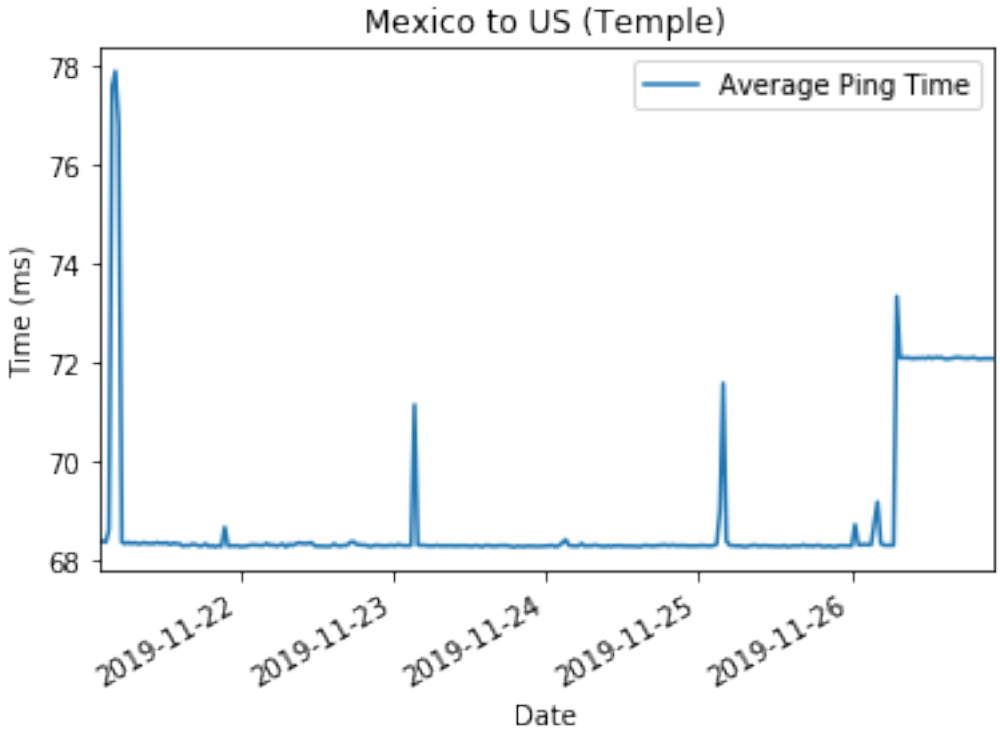
[277]:
```
for title, group in df[df['measure'] == 'ping'].groupby(['src_id', 'dest_id']):
    ax = group.plot(x = 'time', y = 'avg_ping_time', title =␣
 ↪f'{node_loc[title[0]]} to {node_loc[title[1]]}')
    ax.legend(['Average Ping Time'])
    ax.set_xlabel('Date')
    ax.set_ylabel('Time (ms)')
```

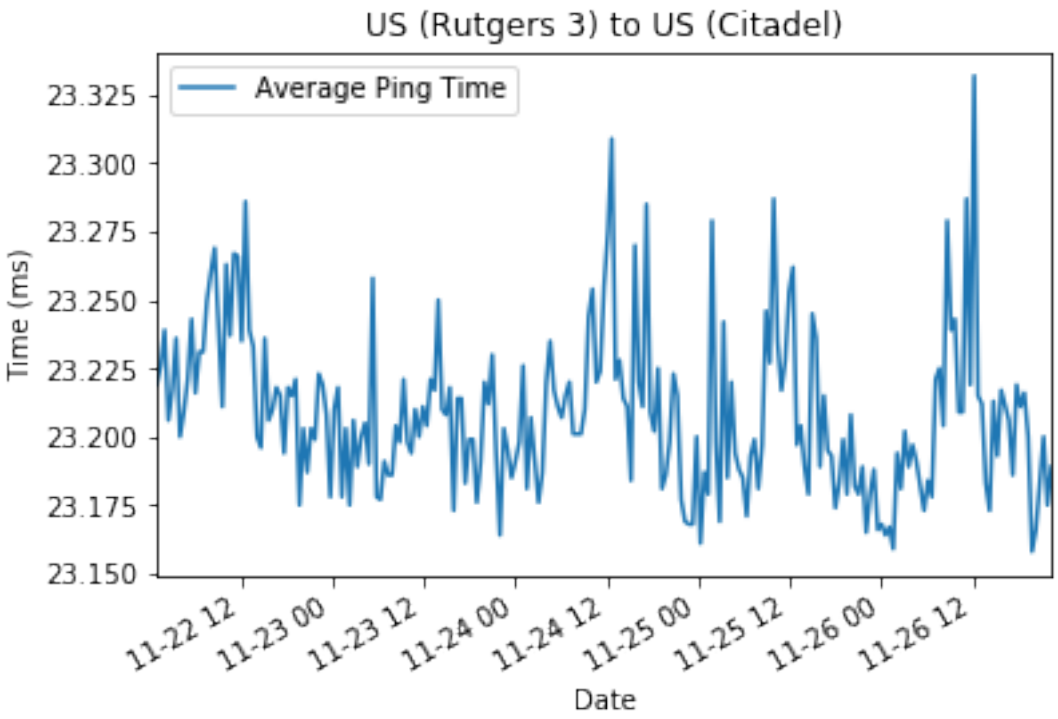


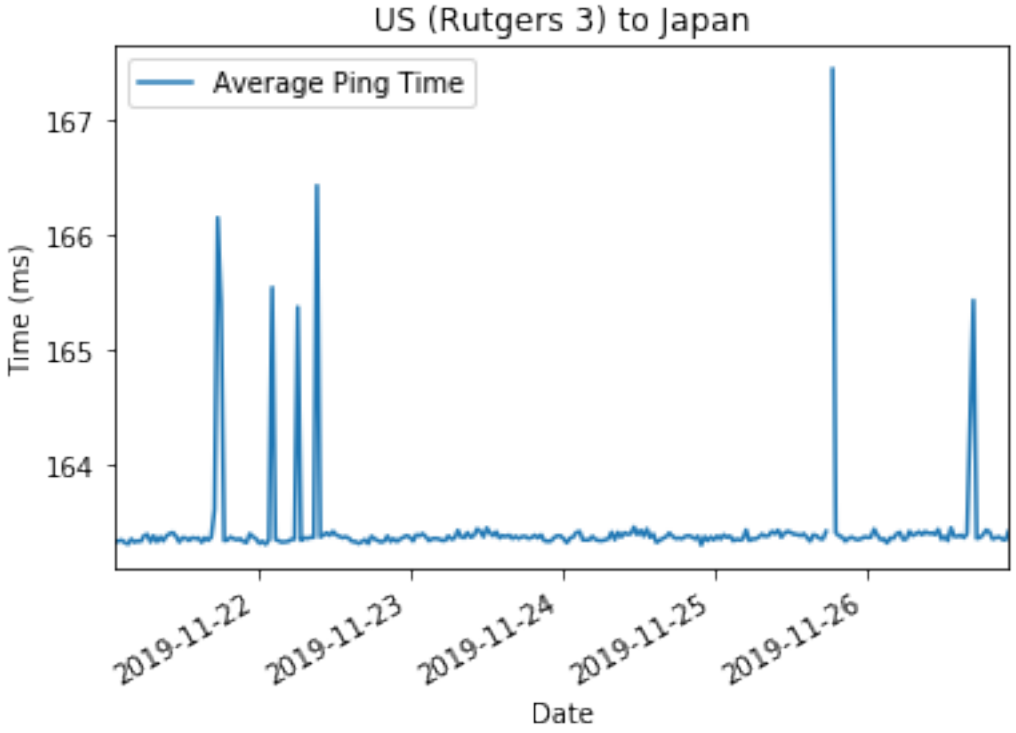

US (Citadel) to US (Rutgers 3)
Average Ping Time
23.40
23.35
23.30
23.25
23.20
Time (ms)
11-22 12
11-23 00
11-23 12
11-24 00
11-24 12
11-25 00
11-25 12
11-26 00
11-26 12
Date

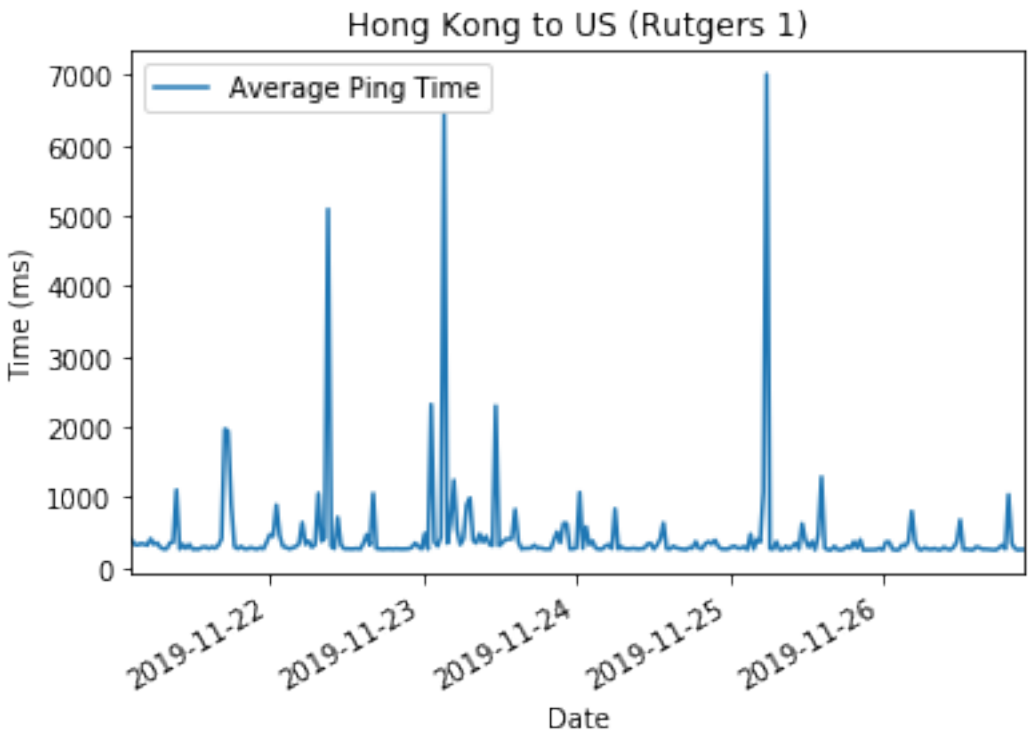

Brazil to US (Citadel)
Average Ping Time
260
240
220
200
180
160
Time (ms)
2019-11-22
2019-11-23
2019-11-24
2019-11-25
2019-11-26
Date

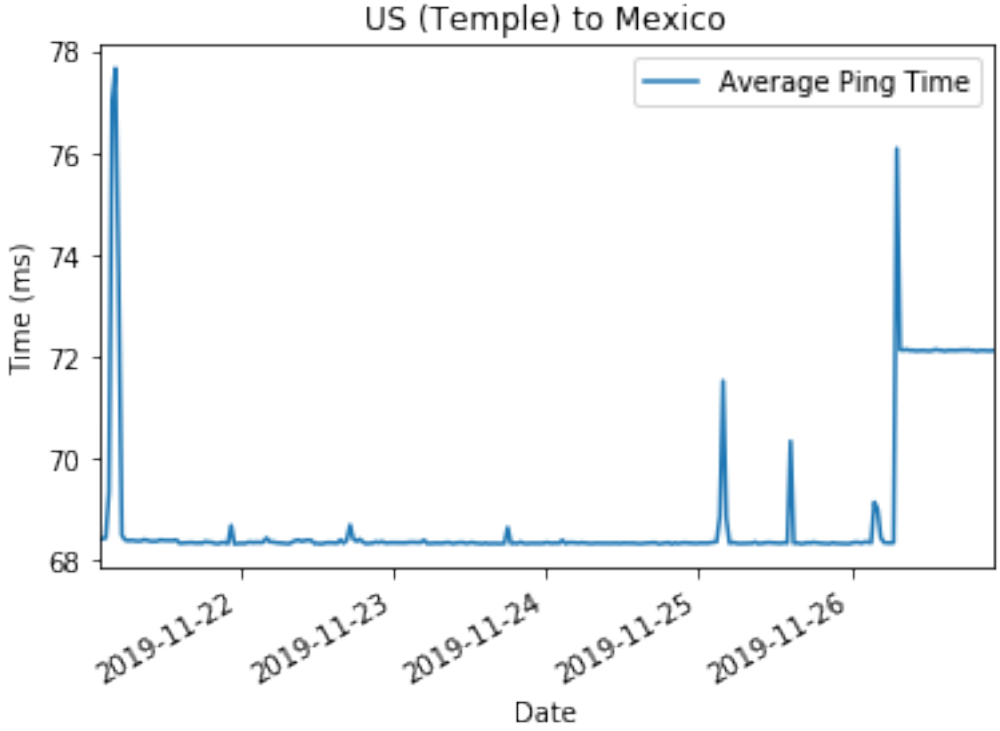

Japan to US (Rutgers 3)
Average Ping Time
Time (ms)
170
169
168
167
166
165
164
163
2019-11-22
2019-11-23
2019-11-24
2019-11-25
2019-11-26
Date

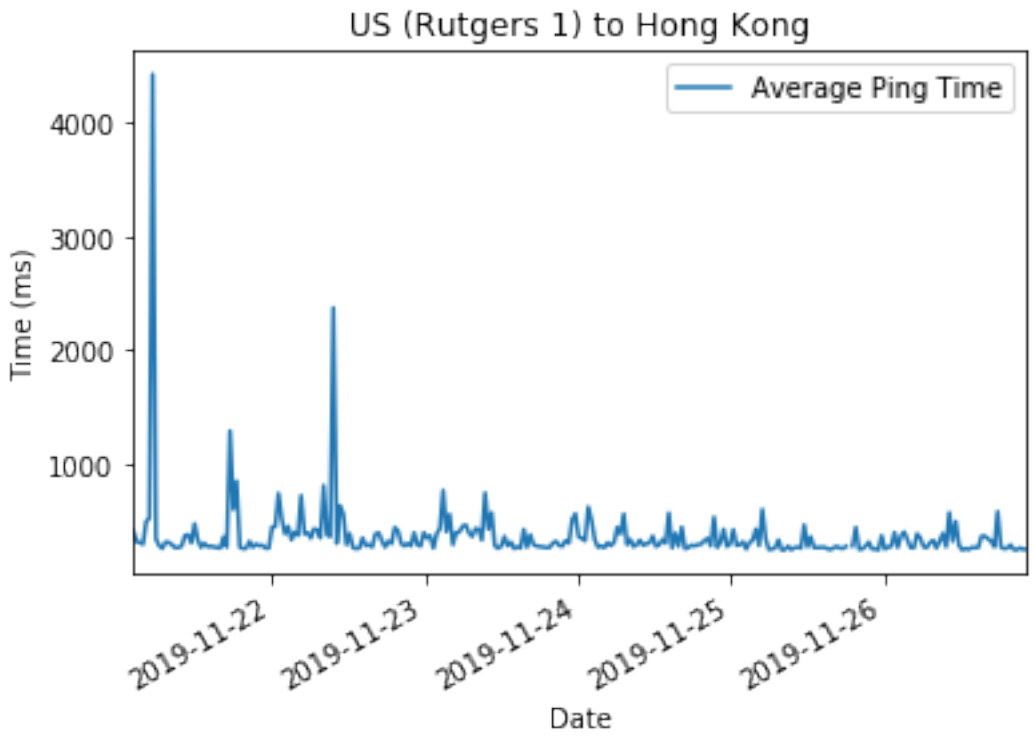

Mexico to US (Temple)
Average Ping Time
Time (ms)
78
76
74
72
70
68
2019-11-22
2019-11-23
2019-11-24
2019-11-25
2019-11-26
Date


US (Rutgers 3) to US (Citadel)
Average Ping Time
Time (ms)
23.325
23.300
23.275
23.250
23.225
23.200
23.175
23.150
11-22 12
11-23 00
11-23 12
11-24 00
11-24 12
11-25 00
11-25 12
11-26 00
11-26 12
Date


US (Rutgers 3) to Japan
Average Ping Time
Time (ms)
167
166
165
164
2019-11-22
2019-11-23
2019-11-24
2019-11-25
2019-11-26
Date


Hong Kong to US (Rutgers 1)
Average Ping Time
Time (ms)
7000
6000
5000
4000
3000
2000
1000
0
2019-11-22
2019-11-23
2019-11-24
2019-11-25
2019-11-26
Date


US (Temple) to Mexico
Average Ping Time
Time (ms)
78
76
74
72
70
68
2019-11-22
2019-11-23
2019-11-24
2019-11-25
2019-11-26
Date

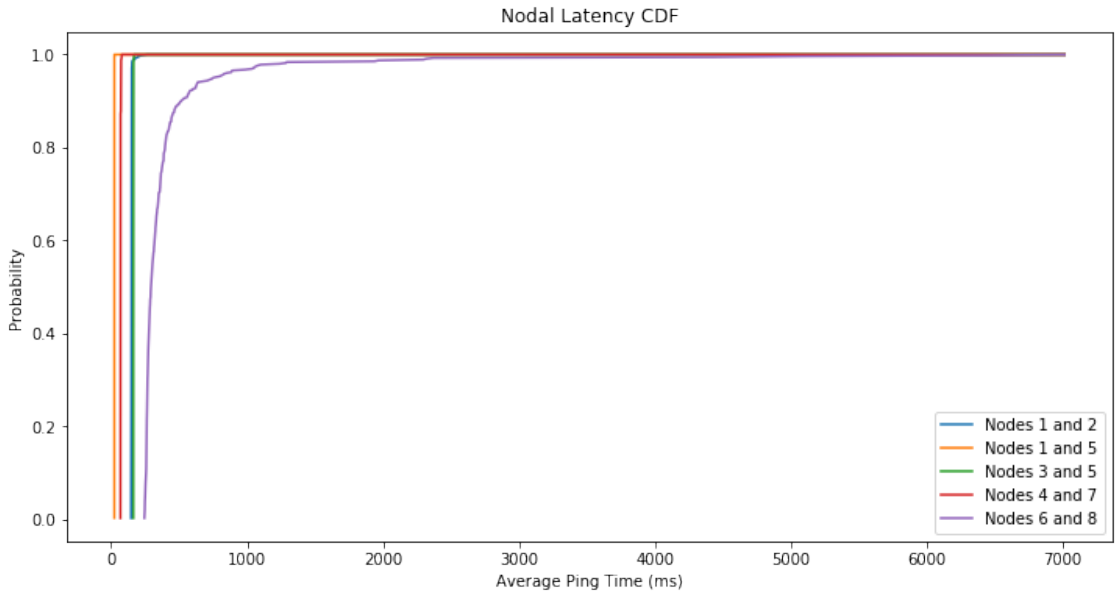


1.0.7 Calculate CDF for packet loss and latency

Create CDF latency and CDF packet loss plots

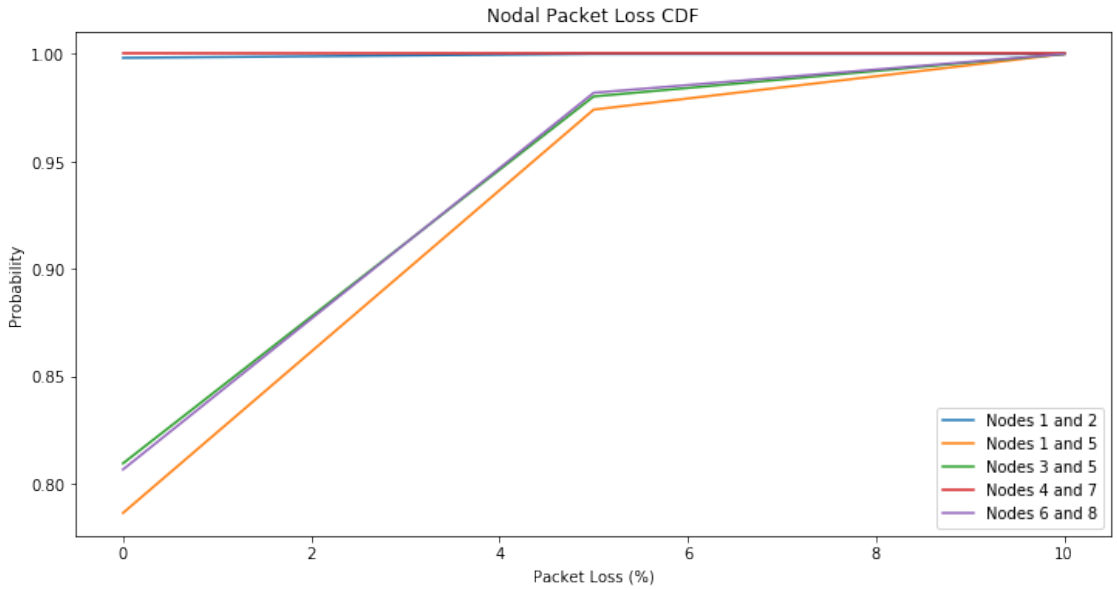
[278]:
```
# first group by each src, desc pair
fig, ax = plt.subplots(figsize=(12, 6))
legend = []
for title, group in df[df['measure'] == 'ping'].groupby(['pair']):

    # then group by the avg_ping_time
    stats_df = group.groupby('avg_ping_time')['avg_ping_time'].agg('count').
↪pipe(pd.DataFrame).rename(
        columns = {'avg_ping_time': 'frequency'})

    # probability that the current time occurs
    stats_df['pdf'] = stats_df['frequency'] / sum(stats_df['frequency'])
    # cumulative probability
    stats_df['cdf'] = stats_df['pdf'].cumsum()
    stats_df = stats_df.reset_index()

    # add the max value to each plot so that the CDF lines continue to end
```

```
    plt.plot(stats_df['avg_ping_time'].tolist()+[df['avg_ping_time'].
↪max()],stats_df['cdf'].tolist()+[1])

    if '1-2' in title:
        legend.append('Nodes 1 and 2')
    elif '1-5' in title:
        legend.append('Nodes 1 and 5')
    elif '3-5' in title:
        legend.append('Nodes 3 and 5')
    elif '4-7' in title:
        legend.append('Nodes 4 and 7')
    elif '6-8' in title:
        legend.append('Nodes 6 and 8')
    ax.legend(legend)
    ax.set_title('Nodal Latency CDF')
    ax.set_xlabel('Average Ping Time (ms)')
    ax.set_ylabel('Probability')
```



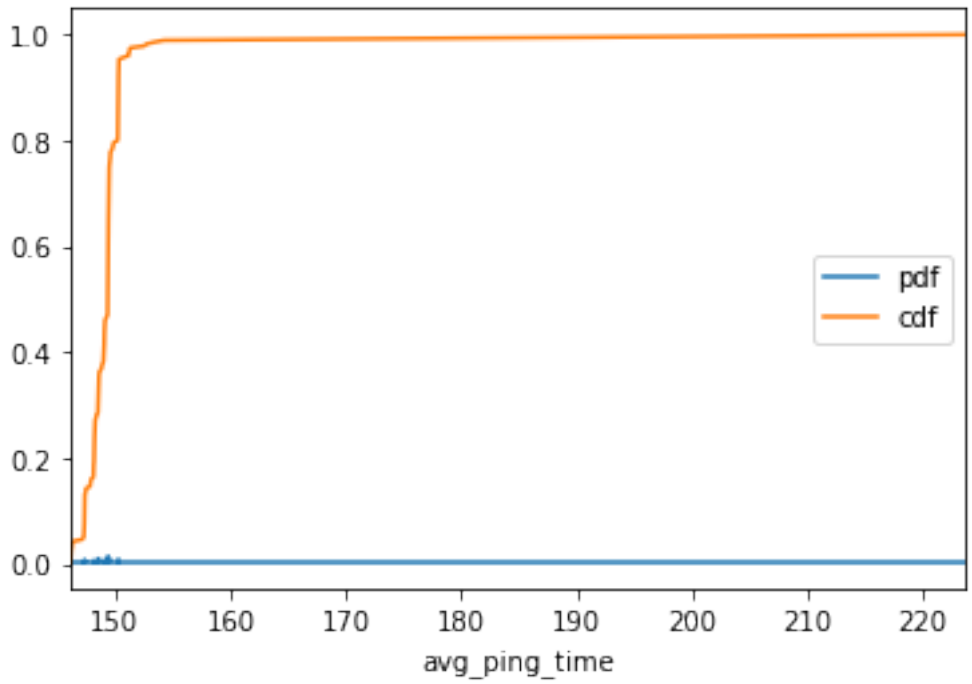
[279]:
```
# first group by each src, desc pair
fig, ax = plt.subplots(figsize=(12, 6))
legend = []
for title, group in df[df['measure'] == 'ping'].groupby(['pair']):

    # then group by the avg_ping_time
    stats_df = group.groupby('packet_loss')['packet_loss'].agg('count').pipe(pd.
↪DataFrame).rename(
        columns = {'packet_loss': 'frequency'})
```

```
    # probability that the current time occurs
    stats_df['pdf'] = stats_df['frequency'] / sum(stats_df['frequency'])
    # cumulative probability
    stats_df['cdf'] = stats_df['pdf'].cumsum()
    stats_df = stats_df.reset_index()

    # add the max value to each plot so that the CDF lines continue to end
    plt.plot(stats_df['packet_loss'].tolist()+[df['packet_loss'].
↪max()],stats_df['cdf'].tolist()+[1])

    if '1-2' in title:
        legend.append('Nodes 1 and 2')
    elif '1-5' in title:
        legend.append('Nodes 1 and 5')
    elif '3-5' in title:
        legend.append('Nodes 3 and 5')
    elif '4-7' in title:
        legend.append('Nodes 4 and 7')
    elif '6-8' in title:
        legend.append('Nodes 6 and 8')
    ax.legend(legend)
    ax.set_title('Nodal Packet Loss CDF')
    ax.set_xlabel('Packet Loss (%)')
    ax.set_ylabel('Probability')
```



[280]:
```
stats_df = df[(df['measure'] == 'ping') & (df['src_id'] == '1') &␣
↪(df['dest_id'] == '2')].groupby(
    'avg_ping_time')['avg_ping_time'].agg('count').pipe(pd.DataFrame).
↪rename(columns = {'avg_ping_time': 'frequency'})

# probability that the current time occurs
stats_df['pdf'] = stats_df['frequency'] / sum(stats_df['frequency'])
# cumulative probability
stats_df['cdf'] = stats_df['pdf'].cumsum()
stats_df = stats_df.reset_index()

stats_df.plot(x = 'avg_ping_time', y = ['pdf', 'cdf'])
```

[280]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x124dd99b0>
```



[283]:
```
df['packet_loss'].sort_values(ascending=False)
```

[283]:
```
1014    10.0
737     10.0
317     10.0
1586    10.0
837     10.0
        ...
3487     0.0
```

```
0        0.0
1279     NaN
1911     NaN
3212     NaN
Name: packet_loss, Length: 2694, dtype: float64
```

[282]:
```
df
```

[282]:
```
      avg_ping_time dest_id                          dest_name  \
0            68.337       7               planetlab1.temple.edu
7           163.325       3  planetlab6.goto.info.waseda.ac.jp
9            23.254       1              planetlab2.citadel.edu
14           68.336       4               pl-dccd-01.cua.uam.mx
15          163.379       5              planetlab3.rutgers.edu
…             …      …                                  …
5400         23.250       5              planetlab3.rutgers.edu
5402        149.697       1              planetlab2.citadel.edu
5403         68.303       7               planetlab1.temple.edu
5405         23.189       1              planetlab2.citadel.edu
5406        150.253       2            planetlab2.c3sl.ufpr.br

                              filename  max_ping_time measure  min_ping_time  \
0     ping_4-7_2019-11-22_06-36-06.txt         68.498    ping         68.244
7     ping_5-3_2019-11-24_00-42-28.txt        163.370    ping        163.271
9     ping_5-1_2019-11-24_10-08-31.txt         23.432    ping         23.162
14    ping_7-4_2019-11-23_12-24-35.txt         68.461    ping         68.276
15    ping_3-5_2019-11-25_16-05-43.txt        163.567    ping        163.280
…                                  …              …     …             …
5400  ping_1-5_2019-11-22_14-09-46.txt         23.478    ping         23.159
5402  ping_2-1_2019-11-25_21-18-43.txt        152.606    ping        149.355
5403  ping_4-7_2019-11-24_11-09-21.txt         68.363    ping         68.238
5405  ping_5-1_2019-11-25_13-56-13.txt         23.290    ping         23.140
5406  ping_1-2_2019-11-23_19-17-37.txt        150.338    ping        150.159

      packet_loss       pair  \
0             0.0  4-7 & 7-4
7             0.0  3-5 & 5-3
9             0.0  1-5 & 5-1
14            0.0  4-7 & 7-4
15            0.0  3-5 & 5-3
…           …        …
5400          0.0  1-5 & 5-1
5402          0.0  1-2 & 2-1
5403          0.0  4-7 & 7-4
5405          0.0  1-5 & 5-1
5406          0.0  1-2 & 2-1
```

```
                                                raw_data  sd_ping_time src_id  \
0     PING planetlab1.temple.edu (129.32.84.160) 56(…         0.129      4
7     PING planetlab6.goto.info.waseda.ac.jp (133.9…        0.478      5
9     PING planetlab2.citadel.edu (155.225.2.72) 56(…         0.193      5
14    PING pl-dccd-01.cua.uam.mx (148.206.185.33) 56…         0.322      7
15    PING planetlab3.rutgers.edu (165.230.49.118) 5…         0.322      3
…                                                   …           …    …
5400  PING planetlab3.rutgers.edu (165.230.49.118) 5…         0.173      1
5402  PING planetlab2.citadel.edu (155.225.2.72) 56(…         0.823      2
5403  PING planetlab1.temple.edu (129.32.84.160) 56(…         0.332      4
5405  PING planetlab2.citadel.edu (155.225.2.72) 56(…         0.202      5
5406  PING planetlab2.c3sl.ufpr.br (200.17.202.195) …         0.391      1

                               src_name                time
0                 pl-dccd-01.cua.uam.mx 2019-11-22 06:36:06
7                planetlab3.rutgers.edu 2019-11-24 00:42:28
9                planetlab3.rutgers.edu 2019-11-24 10:08:31
14                planetlab1.temple.edu 2019-11-23 12:24:35
15    planetlab6.goto.info.waseda.ac.jp 2019-11-25 16:05:43
…                                   …                 …
5400             planetlab2.citadel.edu 2019-11-22 14:09:46
5402            planetlab2.c3sl.ufpr.br 2019-11-25 21:18:43
5403              pl-dccd-01.cua.uam.mx 2019-11-24 11:09:21
5405             planetlab3.rutgers.edu 2019-11-25 13:56:13
5406             planetlab2.citadel.edu 2019-11-23 19:17:37

[2694 rows x 14 columns]
```

[ ]: