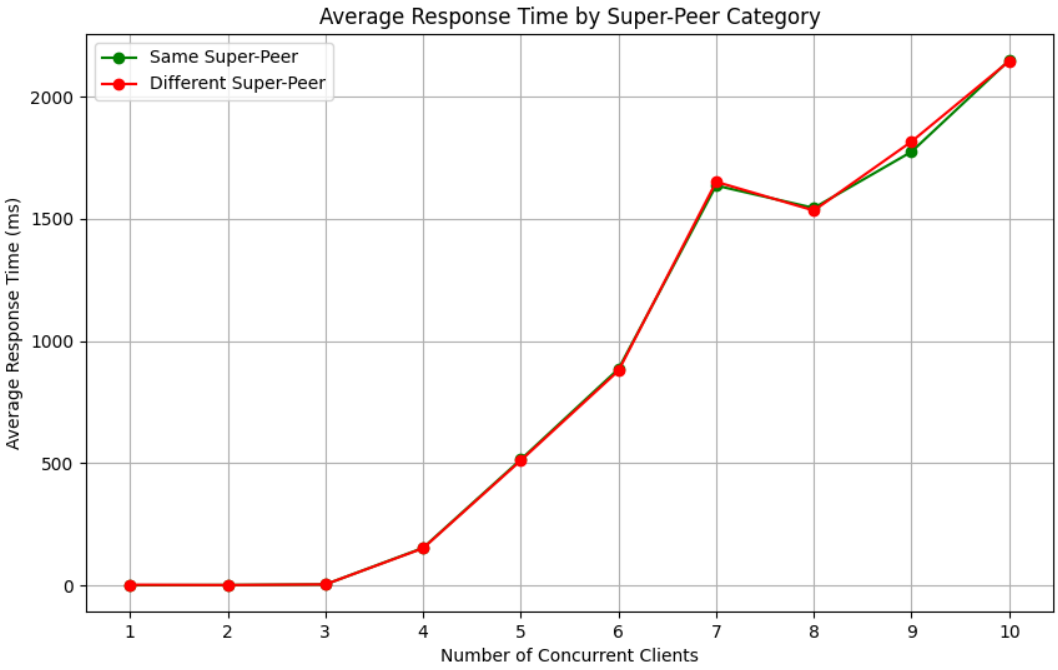
Performance Results of All to All Topology

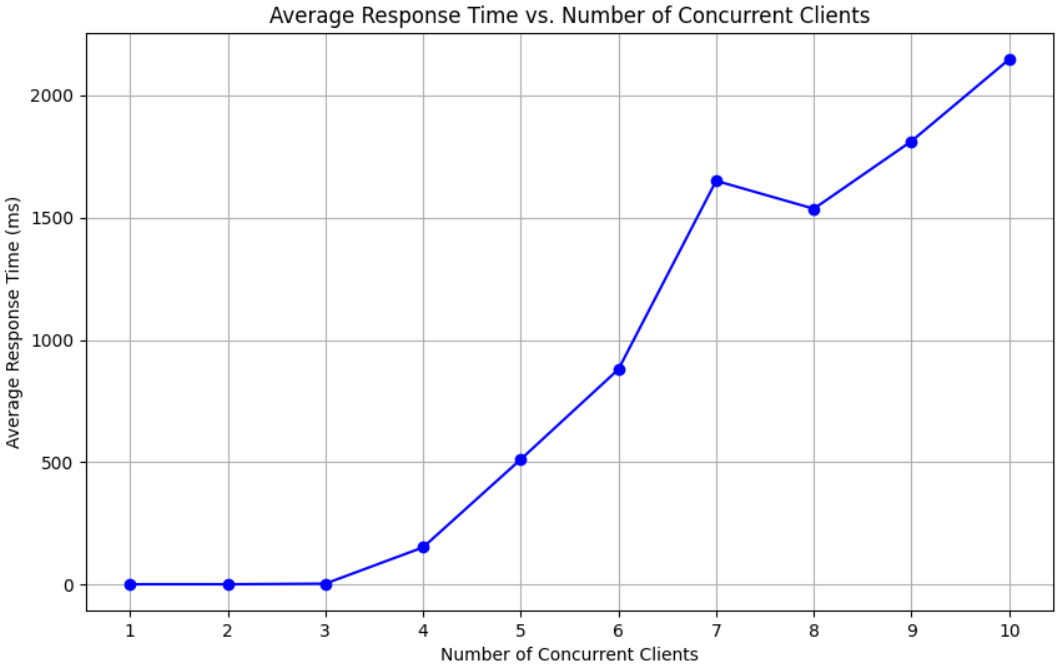
1. **Average Response Time by Super-Peer Category**:
    - **Description**: This plot compares the average response time between two categories: requests handled by the same super-peer and requests handled by different super-peers.
    - **Observation**: As the number of concurrent clients increases, the response time increases for both categories. However, the "Different Super-Peer" category generally has a higher response time than the "Same Super-Peer" category.
    - **Interpretation**: This suggests that inter-super-peer communication adds latency, potentially due to the additional hops or routing complexity involved in reaching a file hosted on a different super-peer.
2. **Response Time vs. Number of Concurrent Clients (Scatter Plot)**:
    - **Description**: This scatter plot shows the distribution of response times for increasing numbers of concurrent clients, categorized by "Same Super-Peer" and "Different Super-Peer."
    - **Observation**: Response times are relatively low and consistent with fewer clients. However, as concurrency increases, response times vary widely, especially for the "Different Super-Peer" category.
    - **Interpretation**: Higher concurrency introduces variability in response times due to network congestion, load on super-peers, or limitations in handling multiple requests simultaneously. The scatter plot emphasizes that performance degrades with concurrency, especially across different super-peers.
3. **Violin Plot of Response Times per Number of Clients**:
    - **Description**: The violin plot visualizes the distribution of response times as the number of clients increases. Wider sections in each "violin" represent higher frequency in response times.
    - **Observation**: With fewer clients, response times are tightly clustered, indicating consistent performance. As the number of clients rises, the distribution widens, reflecting increased response time variability.

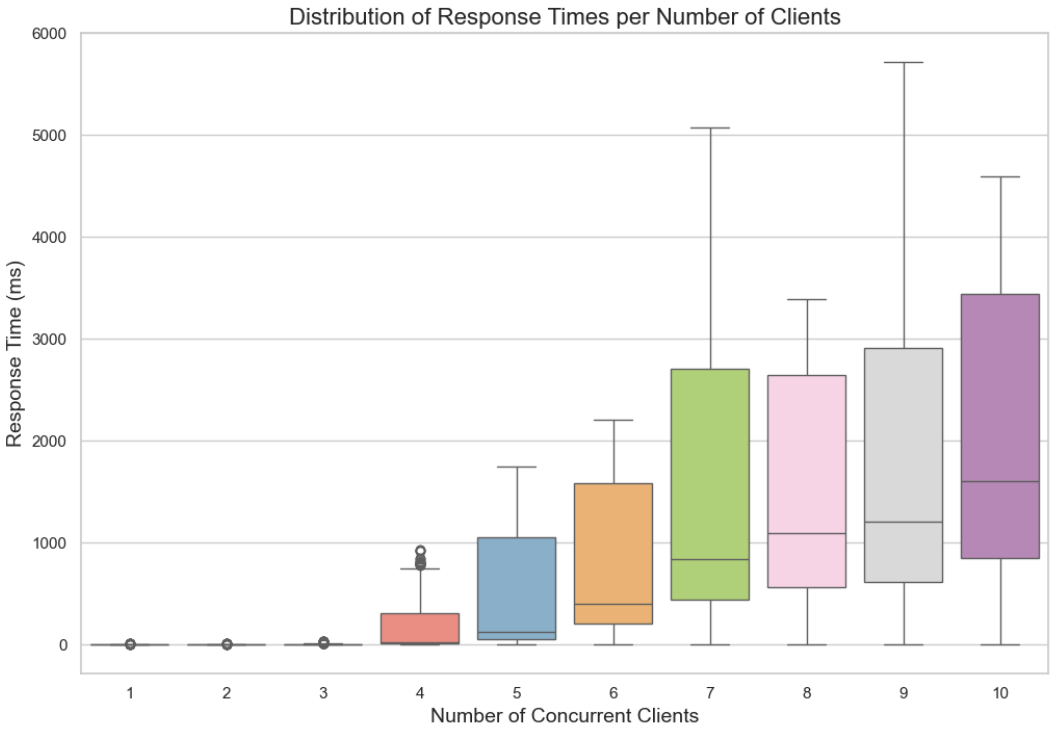
- **Interpretation**: This plot confirms that response times become unpredictable with higher concurrency, particularly beyond 6 clients, due to network and processing bottlenecks.

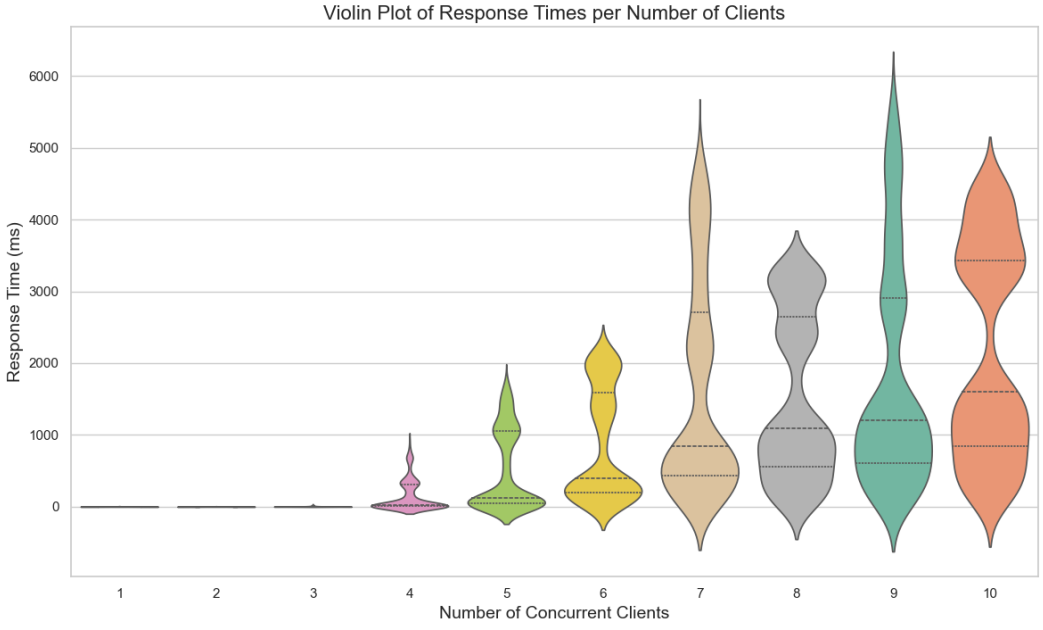
4. **Average Response Time vs. Number of Concurrent Clients (Line Plot)**:
    - **Description**: This line plot illustrates the average response time as a function of the number of concurrent clients, without differentiating by super-peer category.
    - **Observation**: The plot shows an exponential increase in response time with the number of clients, indicating that system performance degrades significantly with higher concurrency.
    - **Interpretation**: The line plot highlights that the system is best suited for lower concurrency levels and struggles to maintain low response times as client count increases, likely due to limited resources or lack of scalability in the current setup.

These plots collectively indicate that the peer-to-peer network handles low client loads efficiently, but faces challenges with high concurrency, especially across different super-peers.


Average Response Time by Super-Peer Category
Same Super-Peer
Different Super-Peer
Average Response Time (ms)
0
500
1000
1500
2000
1
2
3
4
5
6
7
8
9
10
Number of Concurrent Clients


Average Response Time vs. Number of Concurrent Clients
Average Response Time (ms)
0
500
1000
1500
2000
1
2
3
4
5
6
7
8
9
10
Number of Concurrent Clients


Distribution of Response Times per Number of Clients
Response Time (ms)
0
1000
2000
3000
4000
5000
6000
1
2
3
4
5
6
7
8
9
10
Number of Concurrent Clients


Violin Plot of Response Times per Number of Clients
Response Time (ms)
0
1000
2000
3000
4000
5000
6000
1
2
3
4
5
6
7
8
9
10
Number of Concurrent Clients

Average Plot

```
import matplotlib.pyplot as plt
import pandas as pd


def plot_average_response_time():
    # Load summary data
    summary = pd.read_csv('summary_response_times.csv')


    # Plotting
    plt.figure(figsize=(10, 6))
    plt.plot(summary['Number of Clients'], summary['Average Response Time (ms)'], marker='o', linestyle='-', color='b')
    plt.title('Average Response Time vs. Number of Concurrent Clients')
    plt.xlabel('Number of Concurrent Clients')
    plt.ylabel('Average Response Time (ms)')
    plt.grid(True)
    plt.xticks(summary['Number of Clients'])
    plt.savefig('average_response_time_plot.png')
    plt.show()


if __name__ == "__main__":
    plot_average_response_time()
```

boxplot

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

def plot_box_response_time():
    # Load consolidated data
    df = pd.read_csv('consolidated_results.csv')

    # Set the aesthetic style of the plots
    sns.set(style="whitegrid")

    # Initialize the matplotlib figure
    plt.figure(figsize=(12, 8))

    # Create a boxplot
    sns.boxplot(x='Number of Clients', y='response_time_ms', data=df, palette="Set3")

    # Add titles and labels
    plt.title('Distribution of Response Times per Number of Clients', fontsize=16)
    plt.xlabel('Number of Concurrent Clients', fontsize=14)
    plt.ylabel('Response Time (ms)', fontsize=14)

    # Save the plot
    plt.savefig('boxplot_response_time.png')
    plt.show()
```

```
if __name__ == "__main__":
    plot_box_response_time()
```

Scatterplot

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

def plot_scatter_response_time():
    # Load consolidated data
    df = pd.read_csv('consolidated_results.csv')

    # Set the aesthetic style of the plots
    sns.set(style="whitegrid")

    # Initialize the matplotlib figure
    plt.figure(figsize=(14, 8))

    # Create a scatter plot
    scatter = sns.scatterplot(
        x='Number of Clients',
        y='response_time_ms',
        hue='category',
        palette={'Same Super-Peer': 'g', 'Different Super-Peer': 'r'},
        data=df,
        alpha=0.6,
        edgecolor=None
```

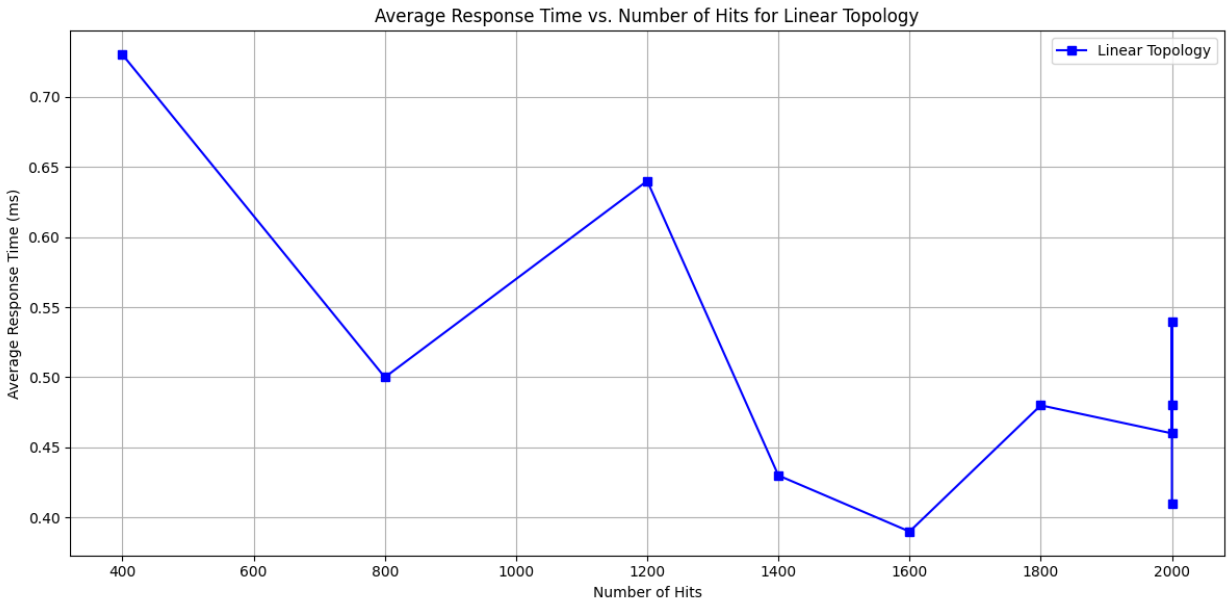
```
    )

    # Add titles and labels
    plt.title('Response Time vs. Number of Concurrent Clients', fontsize=16)
    plt.xlabel('Number of Concurrent Clients', fontsize=14)
    plt.ylabel('Response Time (ms)', fontsize=14)

    # Adjust legend
    plt.legend(title='Category')

    # Save the plot
    plt.savefig('scatterplot_response_time.png')
    plt.show()

if __name__ == "__main__":
    plot_scatter_response_time()
```

Violin Plot

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

def plot_violin_response_time():
    # Load consolidated data
    df = pd.read_csv('consolidated_results.csv')

    # Set the aesthetic style of the plots
```

```
    sns.set(style="whitegrid")

    # Initialize the matplotlib figure
    plt.figure(figsize=(14, 8))

    # Create a violin plot
    sns.violinplot(x='Number of Clients', y='response_time_ms', data=df, palette="Set2", inner='quartile')

    # Add titles and labels
    plt.title('Violin Plot of Response Times per Number of Clients', fontsize=16)
    plt.xlabel('Number of Concurrent Clients', fontsize=14)
    plt.ylabel('Response Time (ms)', fontsize=14)

    # Save the plot
    plt.savefig('violinplot_response_time.png')
    plt.show()

if __name__ == "__main__":
    plot_violin_response_time()
```

Performance of Linear Topology



This plot "Average Response Time vs. Number of Hits for Linear Topology," shows the relationship between the average response time (in milliseconds) and the number of hits (requests) in a linear network topology.

Explanation of Results:

1. **Initial High Response Time:**

   At around 400 hits, the average response time starts high (around 0.7 ms). This could be due to initialization effects, where the network is setting up initial connections or handling a burst of initial requests.

2. **Decrease in Response Time:**

   As the number of hits increases to around 800, the response time drops to approximately 0.5 ms. This might indicate that the network is stabilizing, and connections are becoming more efficient as the system gets "warmed up."

3. **Fluctuations with Higher Hits:**

   The response time fluctuates as the number of hits continues to rise. At around 1200 hits, there's a peak in response time (back to 0.65 ms), followed by a decrease around 1600 hits. These fluctuations may result from periodic network congestion, server load, or the linear topology's inefficiencies in routing data.

4. **Trend towards Stability:**

After 1600 hits, the response times generally stabilize, though there is some variation at the higher end (close to 2000 hits). This indicates that, while the system can handle a moderate increase in hits, the linear topology causes slight inconsistencies in response times as load increases.

Interpretation:

- **Linear Topology Constraints:**

  A linear topology, where nodes are connected in a single line or chain, may suffer from latency issues as more nodes are involved in forwarding requests. This topology is sensitive to load and may lead to varying response times depending on where congestion occurs along the chain.

- **Potential Bottlenecks:**

  The increase and subsequent decrease in response times could indicate that certain nodes are temporarily bottlenecks, especially when requests need to pass through multiple nodes.

This plot demonstrates that while a linear topology can manage moderate load, it is not ideal for high scalability, as fluctuations and peaks in response times may disrupt performance consistency.