Team 6

Emily Hazen
Taylor Carter
Rachel Froling

Train Dispatch

System Set Up

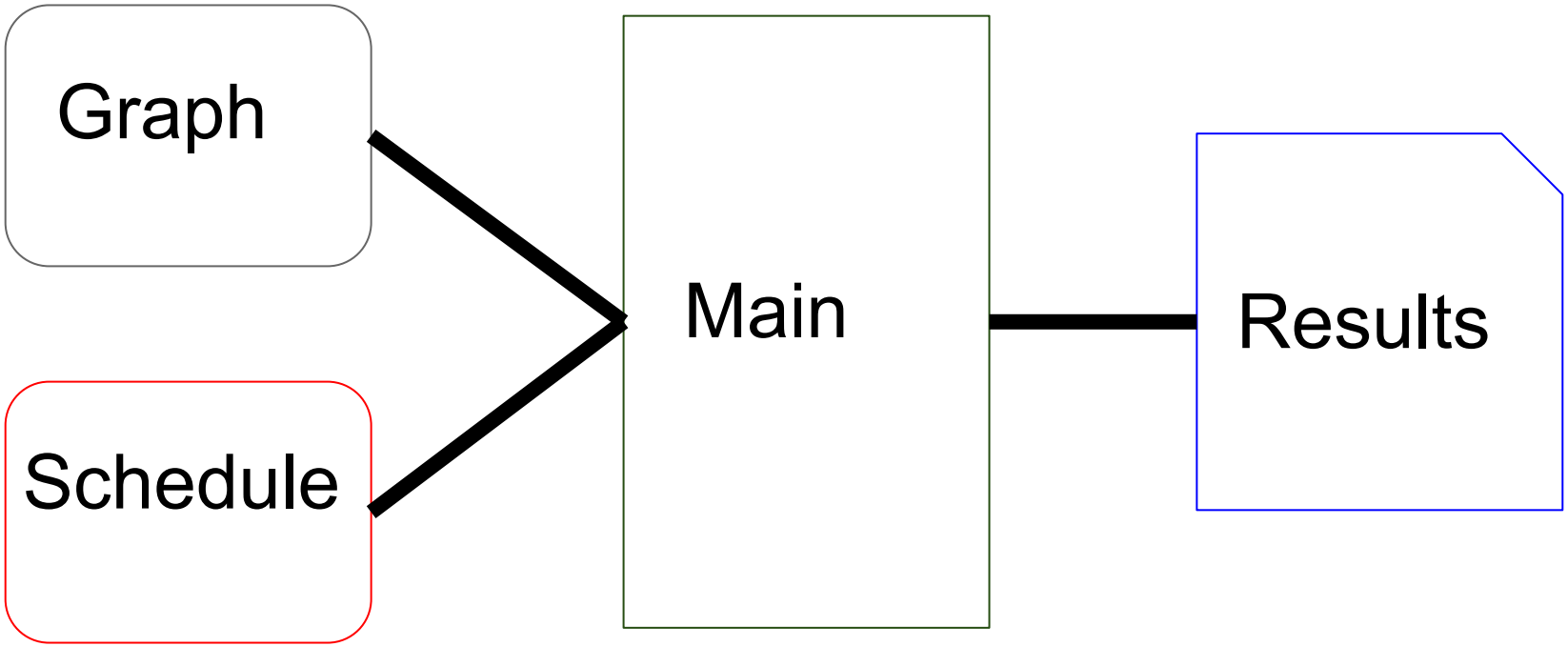
The train dispatch system was designed to dispatch trains in an effective manner.

The system contains three packages:

- Graph - contains all classes relevant to the Multigraph system
- Main - contains all classes for dispatch trains and finding shortest path
- Schedule - contains all classes relevant to scheduling the trains

Train System

- PriorityQueue sorts and polls schedules in order
- Dispatches trains as they become available
- Sprites handle each train’s display and location
  - Sprites are removed when a train arrives at its destination
- Weight of an edge is the speed at which a train (sprite) travels on it
- Cost = time train arrives at destination - time train was ready
- Program ends when there are no more trains to dispatch and all trains have arrived at destination



Input & Output

Baseline and Improvement

Baseline

- Basic single-source shortest path that skips any locked edge
- Schedule will not be polled until an edge is found
    - It will skip these schedules until a path opens up
- All edges in a path are locked at dispatch
- Edges do not unlock until train arrives at destination

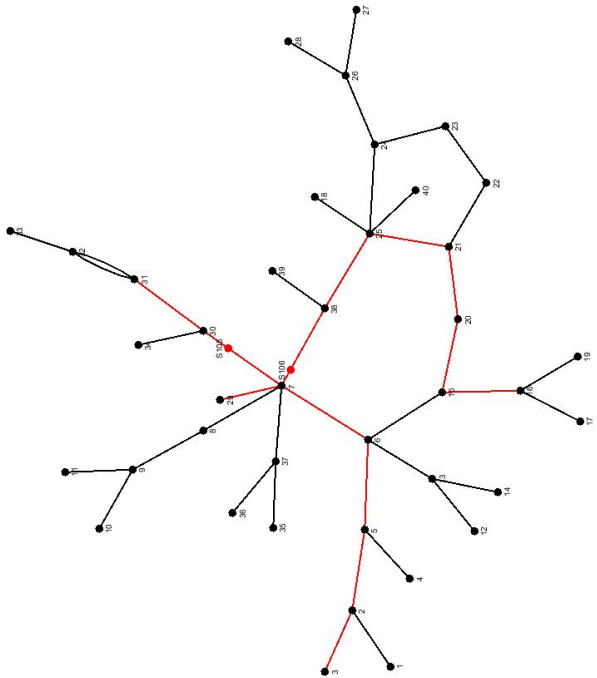
Improvement

- Add a reservation system to track when a path is going to be in use
- Amount of time spent waiting for an edge to open considered in the cost of using that edge.
- Determines if it is more cost-effective to wait for a path to open up or to use a path that is available immediately.
- Once a path is chosen, each edge is reserved for the time at which the train will be using it
    - Determined by time the train arrives at the edge & time at which the edge is available
- A train only locks the edge it is using

Reservation System

- A reservation list is attributed to every edge
- The availability of an edge is determined by these reservations
  - If the edge is reserved at the time a train needs it, the next time the edge is available is found and attributed to the edge
- Edge delay = availability - arrival time of the train
  - Dijkstra’s adds this delay to the cost of using that edge
- Trains keep track of their own reservations to know when they can use the next edge
  - If a train has to wait at a station, it is marked as delayed

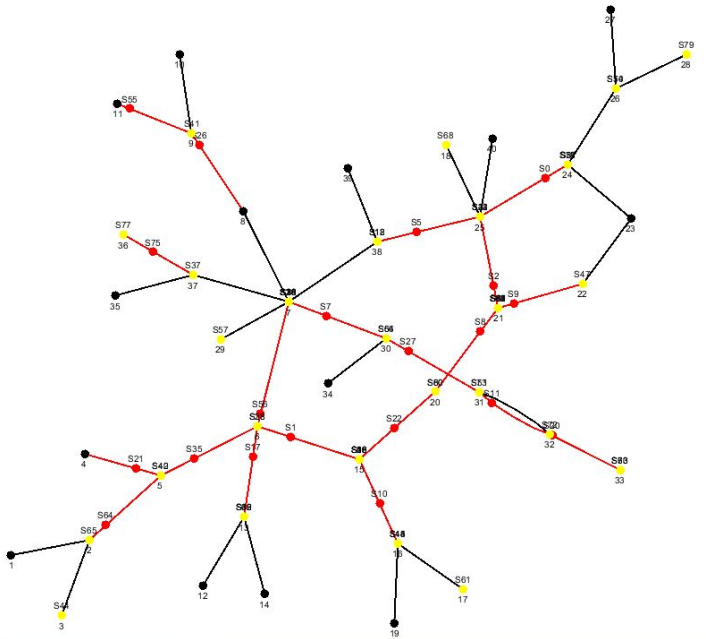
Example run: Paris



Base Case

Red dots represent moving trains

Red edges are locked



Improvements

Yellow dots represent delayed trains

Analysis

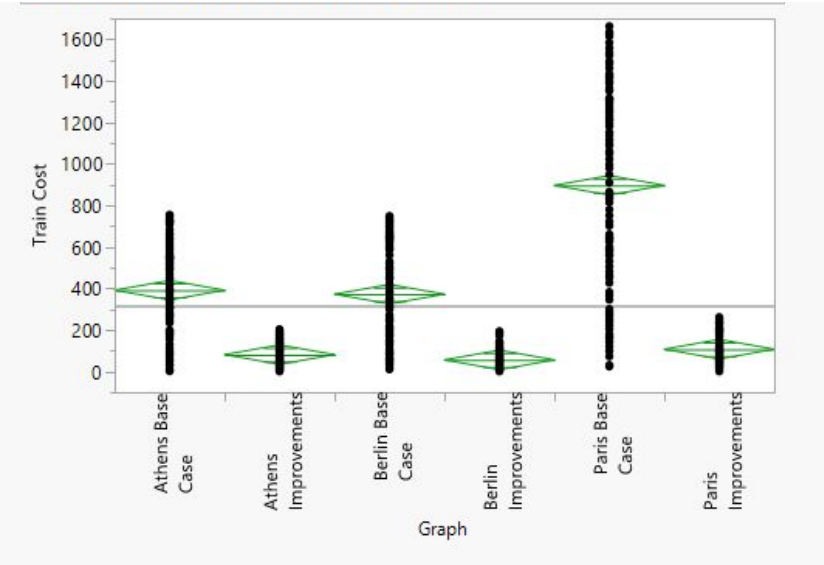
Testing and Analysis

To test if our improvement optimized the cost of sending a schedule of trains:

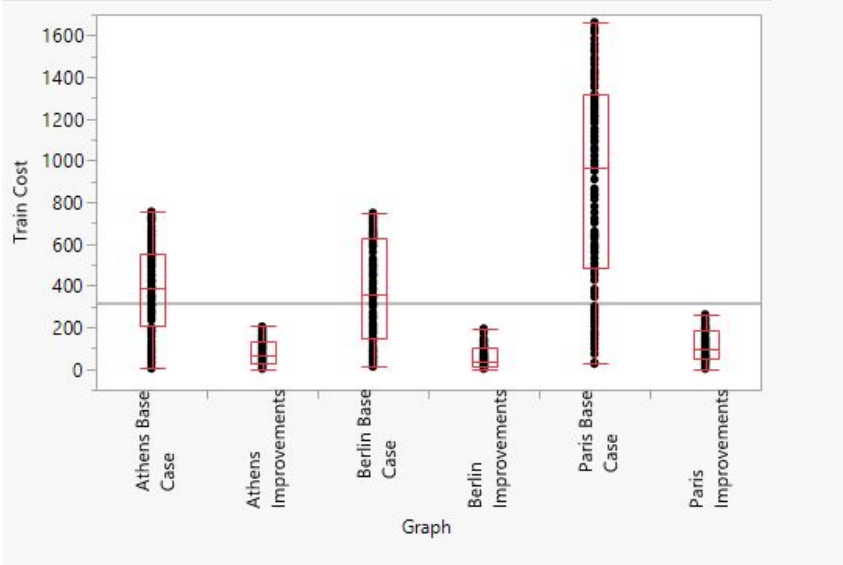
- Tested the density of when trains were dispatched
- Tested different maps
- Tested different number of trains

100 trains in 25 time intervals

ANOVA:

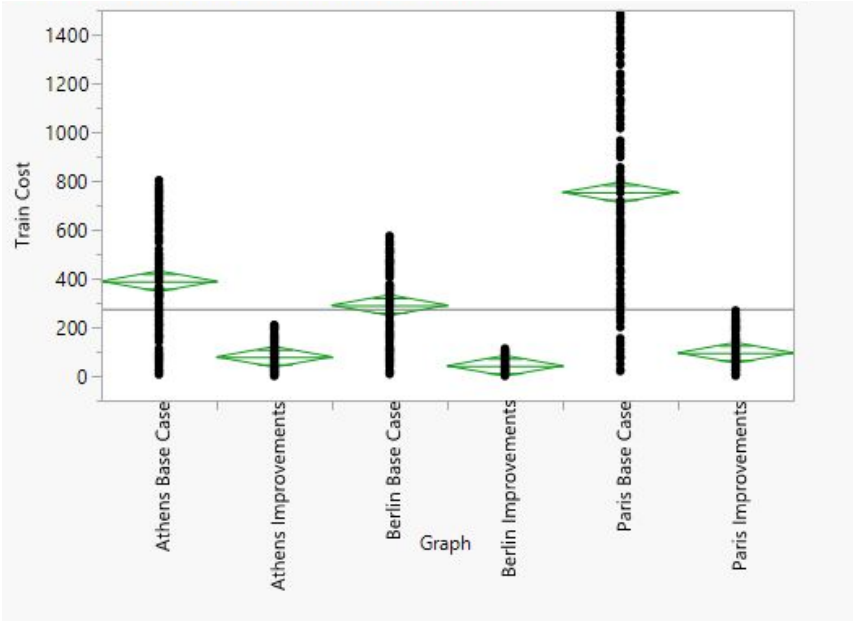


Box and Whisker:

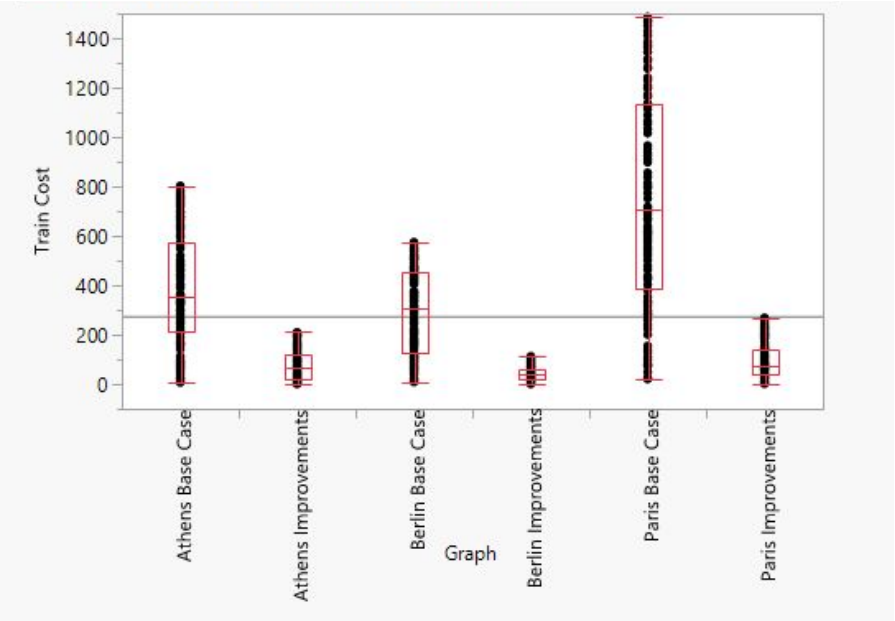


100 trains in 50 time intervals

ANOVA:

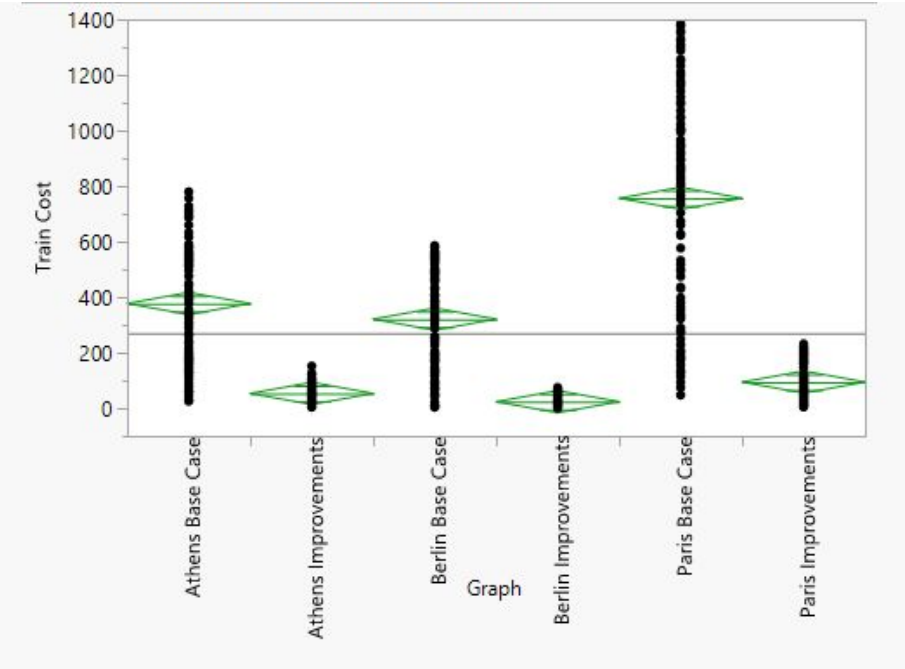


Box and Whisker:

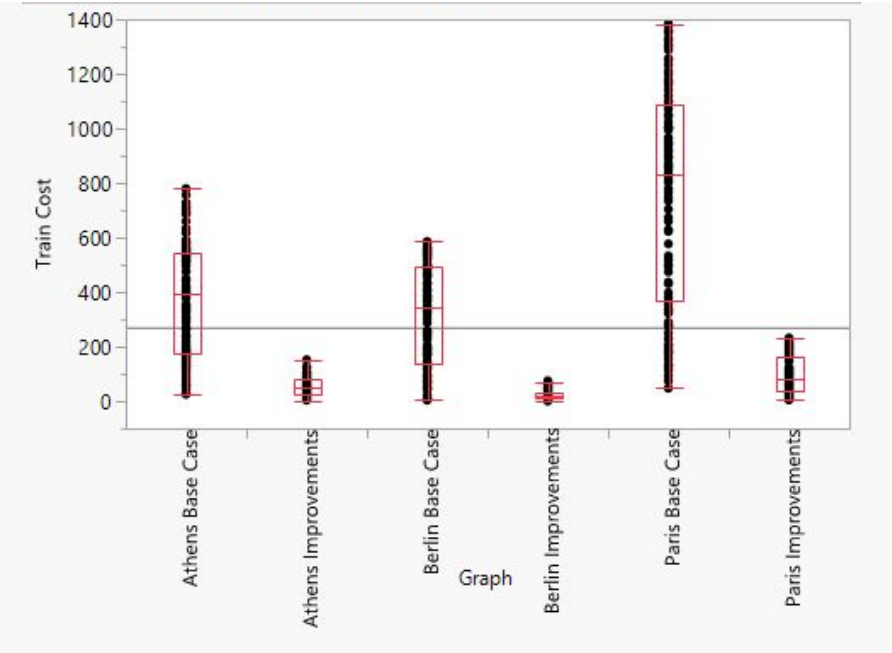


100 trains in 100 time intervals

ANOVA:

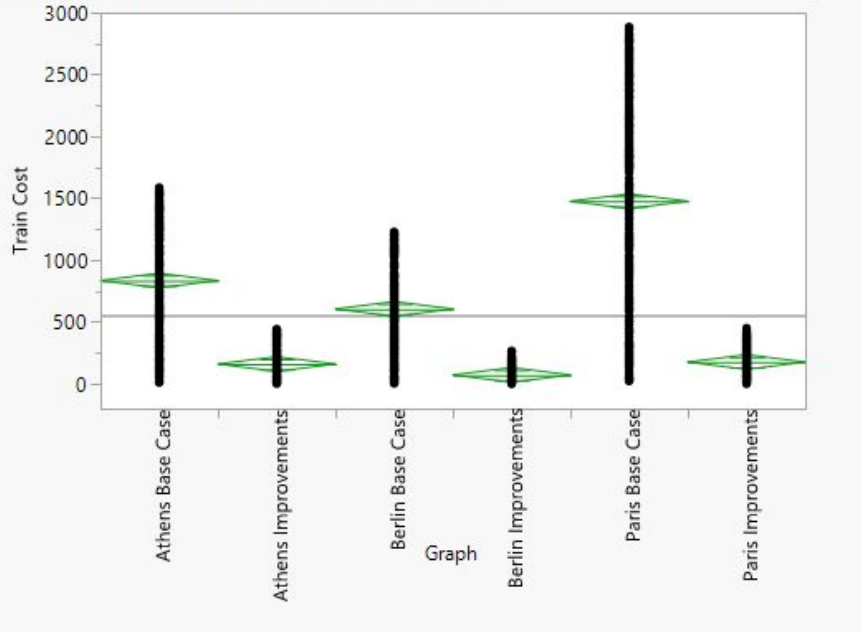


Box and Whisker:

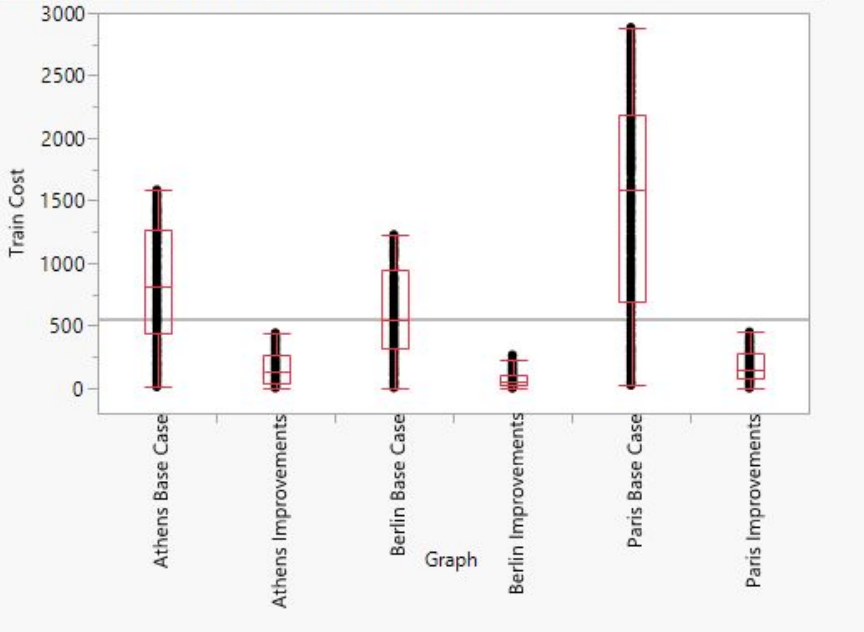


200 trains in 25 time intervals

ANOVA:

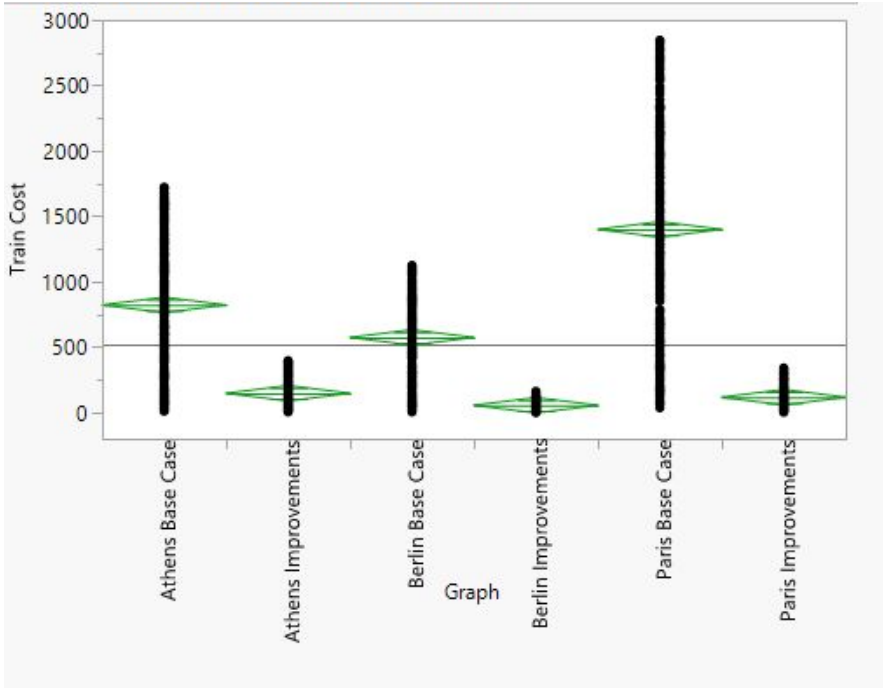


Box and Whisker:

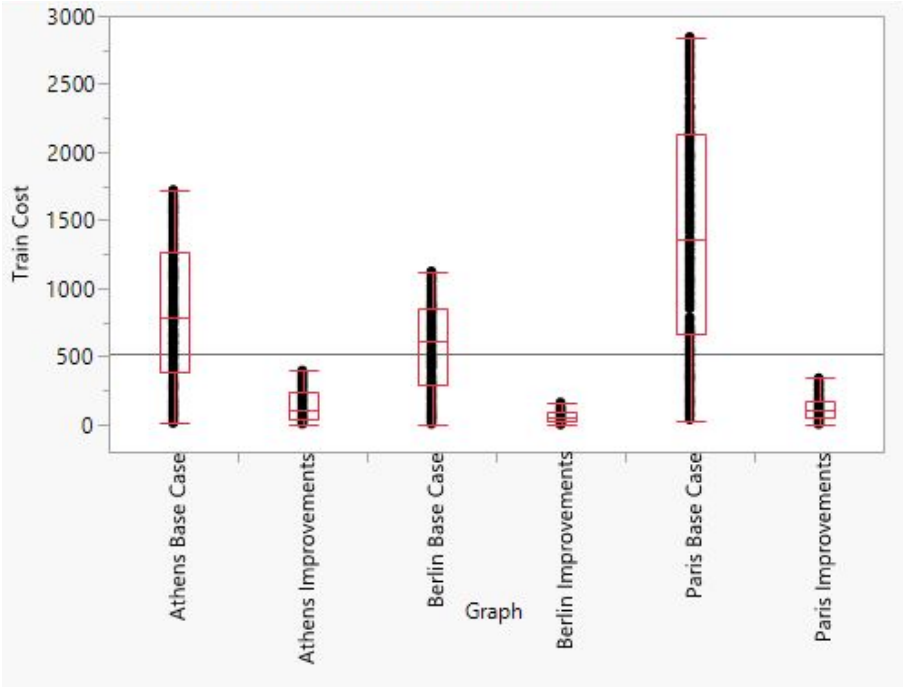


200 trains in 50 time intervals

ANOVA:

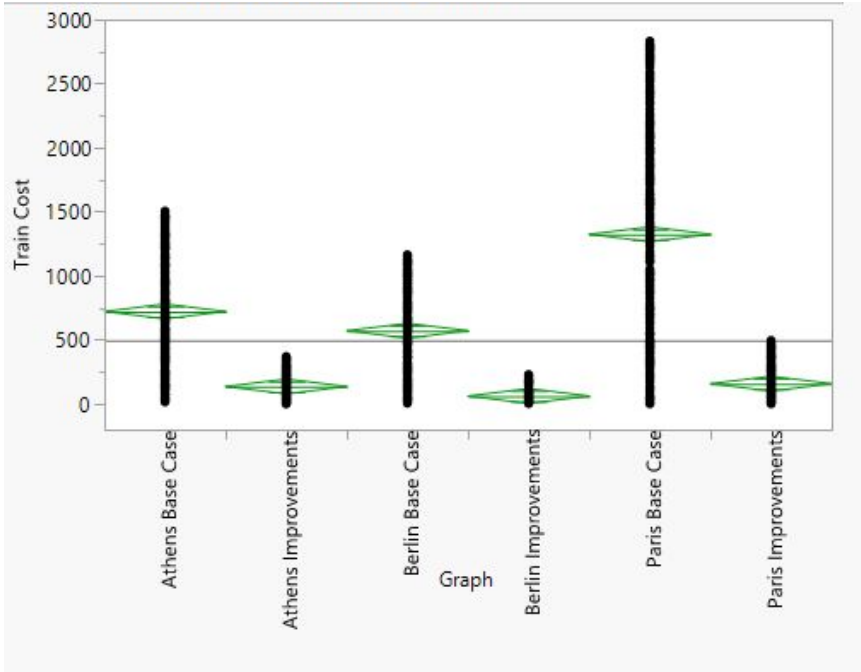


Box and Whisker:



200 trains in 100 time intervals

ANOVA:



Box and Whisker: