"Learning to Route"
Restoring Results

Report

Written by:

Ido Yehezkel 204397368

Table of Contents

Introduction

This report is about restoring the achieved results from the paper: Valadarsky, A., Schapira, M., Shahaf, D., & Tamar, A. (2017, November). Learning to route. In *Proceedings of the 16th ACM workshop on hot topics in networks* (pp. 185-191).

The paper's writers interduce a new approach of how to use machine learning techniques in order to solve one of the fundamental network control problem, traffic routing.
Two approaches were evaluated: traffic patterns predictions based on the history traffic using supervised learning and solving the routing control problem directly using reinforcement learning agent.
The supervised learning models is a good start point but didn't achieve noticeable results, on the other hand, the reinforcement learning techniques achieved much better results and create new direction of research.

This report is focusing on restoring the paper results and conclusion made using reinforcement learning techniques, also examine how these techniques can be used on other network topologies.

Definition of the Model

Network topology- following the paper, for the control routing problem the network topology is represent by directional graph which is based on unidirectional graph with edges' capacity function (in Mb). The directional graph is simply made by considering each edge as independent bidirectional link.

Synthetic Traffic Generation – to examine the developed techniques a synthetic traffic generation is necessary.

The traffic demands is represent by square matrix where the cell i, j is the flow demand from node i to node j .Another property of the matrix is sparsity, which is the percent of source, destination pairs that include traffic.
Two ways are been used for generating the traffic:

- Gravity: traffic that correlated with the capacities connected to the source node and destination node, calculated by:
  $$\frac{[\text{source out links' capacity}] \times [\text{destination out links' capacity}]}{\text{total nodes out links' capacity}}$$
- Bimodal: the traffic is sampled by some probability (biased coin flip) from two independent gaussian distributions. One distribution represents mice flows and the other elephants' flows.

Optimal Routing – Baseline

To evaluate the models and techniques that were developed by the paper's writer they define a reference baseline which is based on **load balancing**.

The optimal routing criteria is defined as the **minimization of maximum link utilization** (minimize the most congested link), the mathematic expression is:

$$\min_{e \in E} \left\{ \max \frac{f_e}{C_e} \right\}$$

C_e – capacity of link e

f_e – total flow in link e

These criteria can be formulated as an optimization problem as follow:

$$Objective: Minimize\ r$$

$$\sum_i \sum_{j \neq i} g_e(i, j) \leq C_e \cdot r \ \ \forall e \in Edges$$

where $g_e(i, j)$ is a fraction of demened $i \rightarrow j$ flows on link e

The routing scheme constartions:

For each existing demend $i \rightarrow j$:

$$\sum_{e \in IN(v)} g_e(i,v) - \sum_{e \in OUT(v)} g_e(i,v) = demend(i,v) \Big| v = j, \text{destination constraint}$$

$$\sum_{e \in OUT(v)} g_e(v,j) - \sum_{e \in IN(v)} g_e(v,j) = demend(v,j) \Big| v = i, \text{source constraint}$$

$$\sum_{e \in OUT(v)} g_e(i,j) - \sum_{e \in IN(v)} g_e(i,j) = 0 \Big| v \neq i, j, \text{transit constraint}$$

$$g_e(i, j) \geq 0 \ \forall i, j$$

$$r \geq 0$$

As this optimization problem define, all the constraints are linear expressions therefore, a linear programing solver (like "IBM – CPLEX" or "Gurobi") can be used to solve it. The linear programming problem includes $O\left(|edges| \times |nodes|^2\right)$ variables and $O\left(|edges| + |nodes|^2\right)$ constrains.

Optimal Oblivious Routing - Baseline

Another baseline the writers used is **optimal oblivious routing**. As its name implies the oblivious routing is not traffic patterns depended but only a topology depended, this routing technique was represented in several papers 15 years ago.

The oblivious performance ratio of routing scheme f is define as follows:

$$CONGESTION(f,D) = \max_{e\in E} \frac{Flow(e,f,D)}{C_e}\Bigg|$$

Most congestion edge with repect to routing scheme f, demend and capacity.

$$OBLIV-PERF-RATIO(f) = \sup_D \frac{CONGESTION(f,D)}{OPT(D)}\Bigg|$$

Worst demend matrix congestion per routing scheme f normalized with optimal routing.

$$OBLIVE-OPT(G) = \min_f OBLIV-PERF-RATIO(f)\Big|$$

Best routing scheme f of topology G regrdless the traffic demend

Using the result of the paper *Applegate, D., & Cohen, E. (2006). Making routing robust to changing traffic demands: algorithms and evaluation. IEEE/ACM Transactions on Networking, 14(6), 1193-1206,* the optimal oblivious routing problem can be formulated as optimization problem as follow:

$Objective : Minimize\ r$

f is valid routing scheme

$\forall edges\ e:$

$$\sum_{h\in E} C_h \cdot \pi_e(h) \le r$$

$$\forall \text{ pairs } i \to j: \ \frac{f_e(i,j)}{C_e} \le p_e(i,j)$$

$$\forall \text{ node } i,\ \forall \text{ edge a}=(j,k):$$

$$\pi_e(a) + p_e(i,j) - p_e(i,k) \ge 0$$

$$\forall \text{ edge h} \in \text{E}: \pi_e(h) \ge 0$$

$$\forall \text{ node } i: \ p_e(i,i) = 0$$

$$\forall \text{ node } i,j: \ p_e(i,j) \ge 0$$

This also a linear programming problem with $O\left(|edges| \times |nodes|^2\right)$ variables and $O\left(|edges|^2 \times |nodes|\right)$ constrains, where $\pi_e(h)$ is a variable that represent an exist weight that for every pair of edges e,h: $\sum_{h\in E} C_h \cdot \pi_e(h) \le r$ and $p_e(i,j)$ represent the length of the shortest path from node i to node j according the edge weights $\pi_e(h)$.

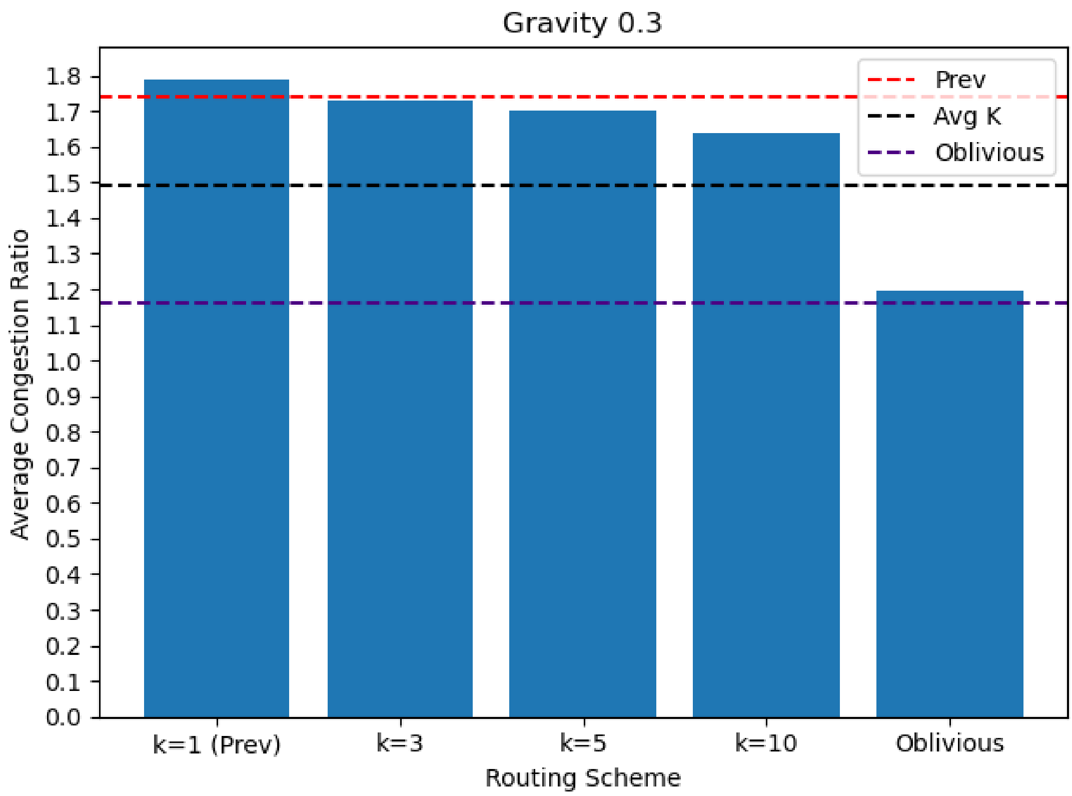
Restoring the Baseline Results

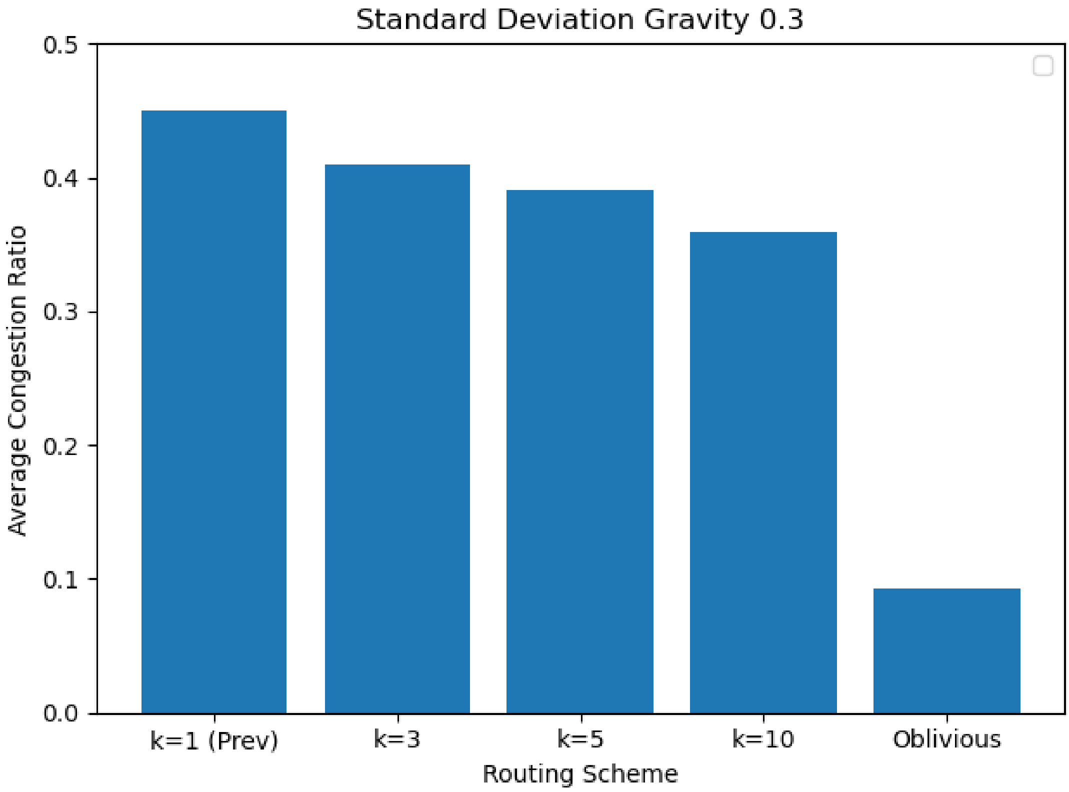
Before restoring the results of the reinforcement learning agent all the baselines should be restored. Because the agent is taking a decision by observing the traffic history the writers created a similar reference baseline that also use traffic history, by observing the last K matrices and calculate the average traffic matrix and route the next new traffic matrix by the optimal routing scheme of the average one.
Another challenge we need to consider is that there is a possibility that a flow is exist in the new matrix but not in the average one, in order to handle that the writers used an ECMP policy with equal weights, so those flows are equally divided between all shortest paths between the source and destination.
Note, the result are normalized with the most congested link utilization when applying the optimal routing scheme.

Evaluation

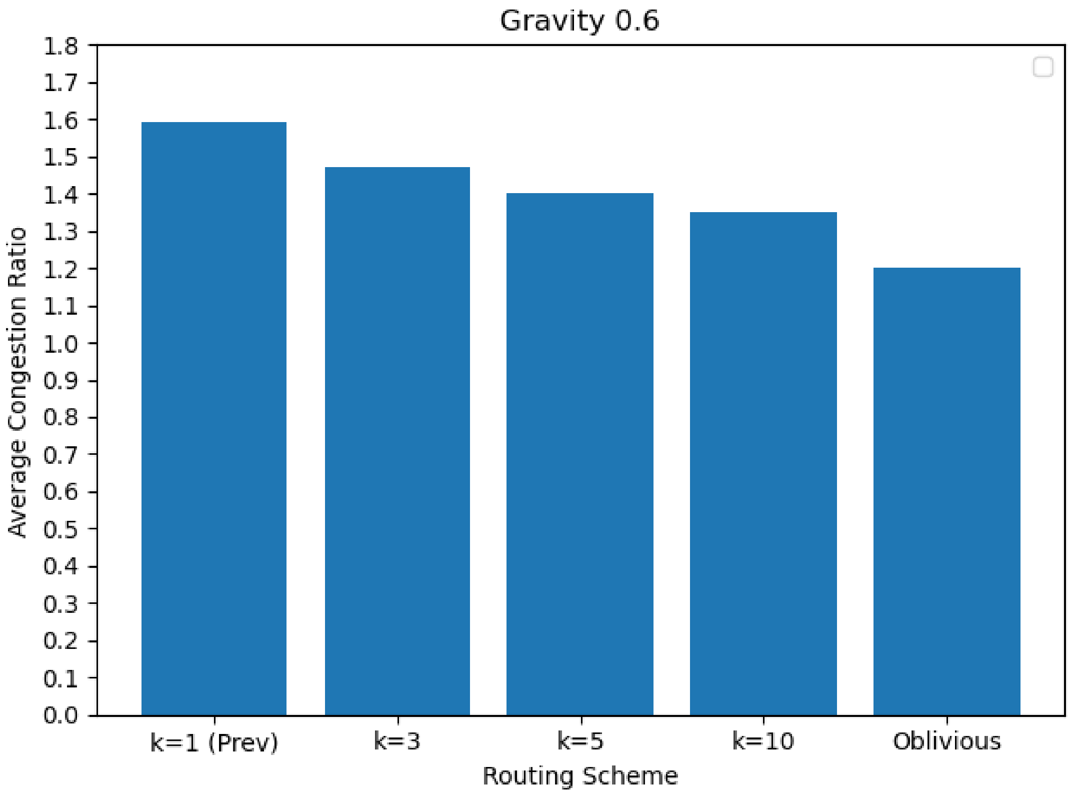
Similar to the paper, a 12-node topology with 26 edges (the original topology includes duplicate edges, this represented as double the capacity for those edges) and constant link capacity of 10,000 Mb.
20,000 traffic matrices dataset has been used in order to get the results.

Gravity Traffic:

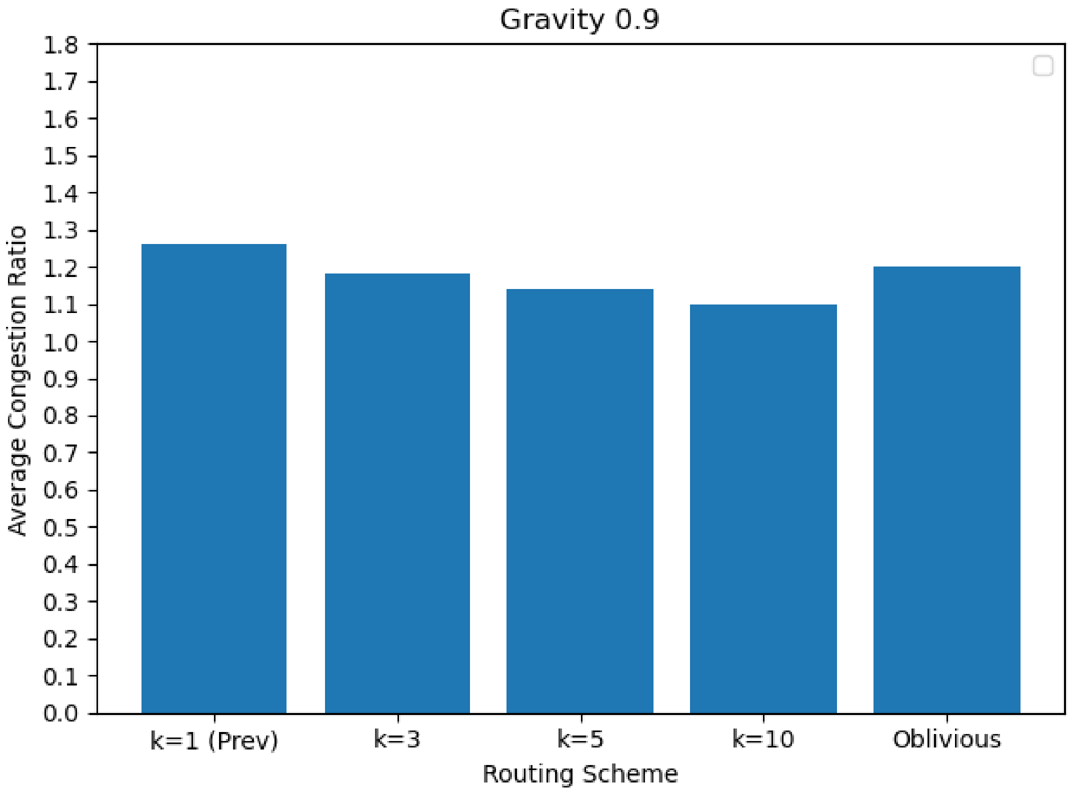




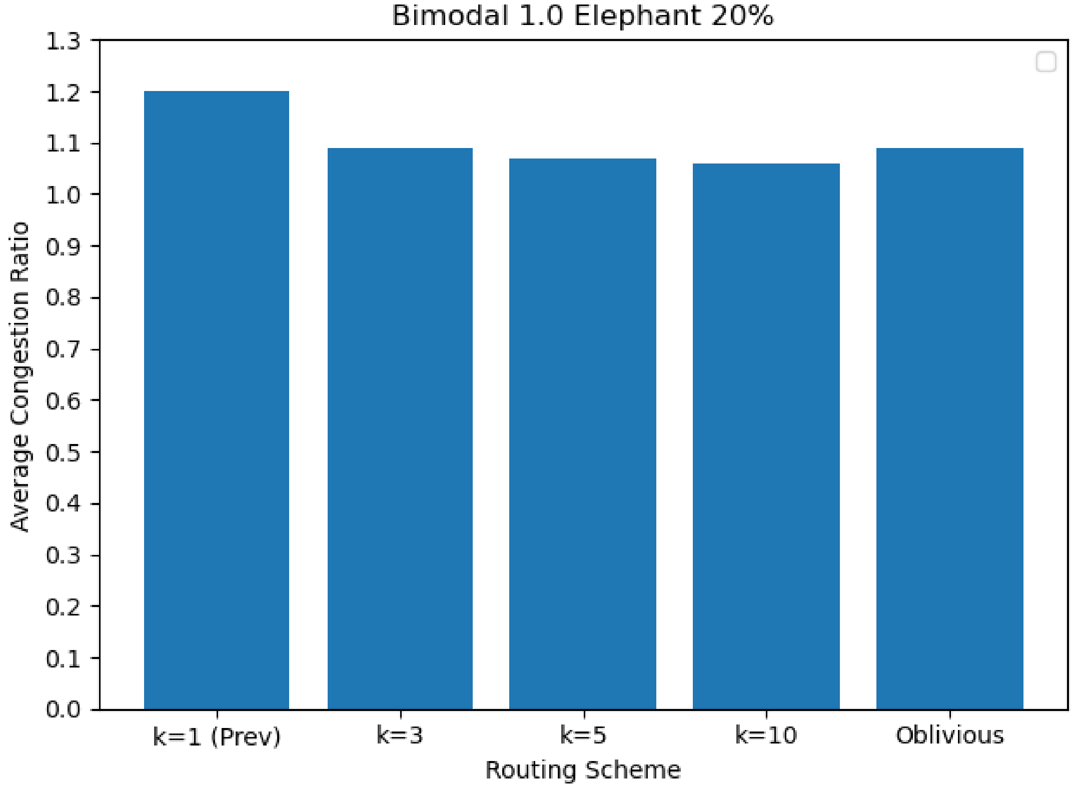
Standard Deviation Gravity 0.3
Average Congestion Ratio
0.0
0.1
0.2
0.3
0.4
0.5
k=1 (Prev)
k=3
k=5
k=10
Oblivious
Routing Scheme

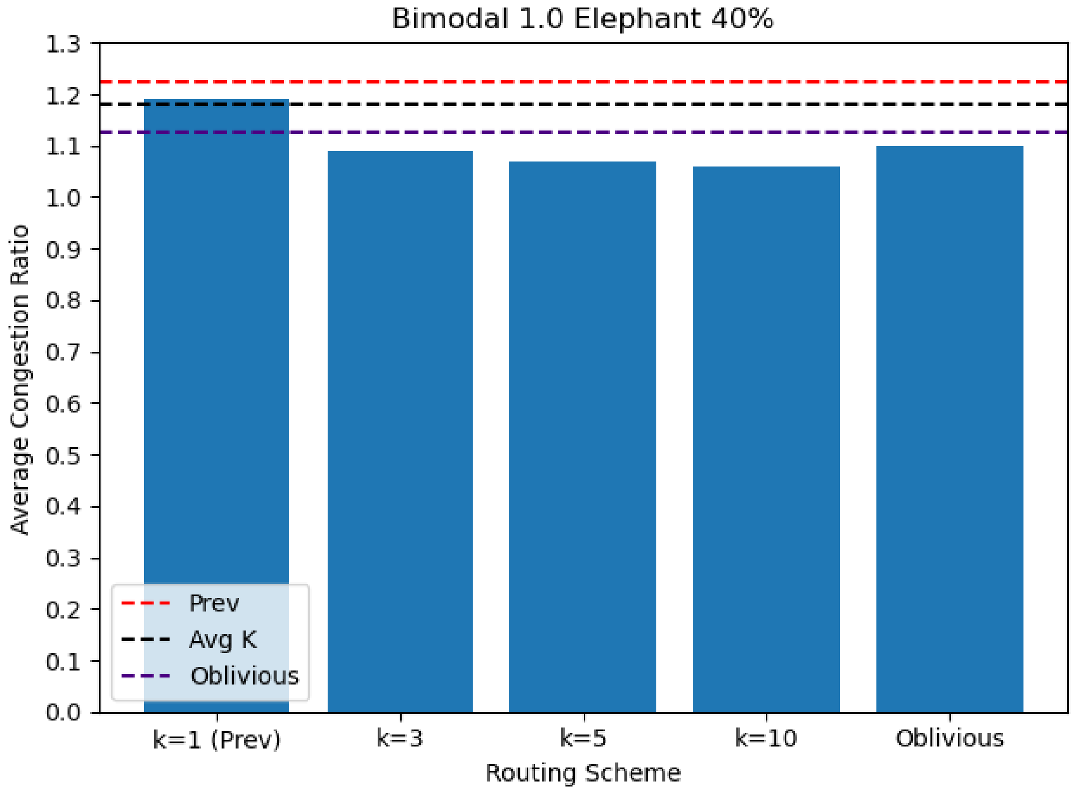


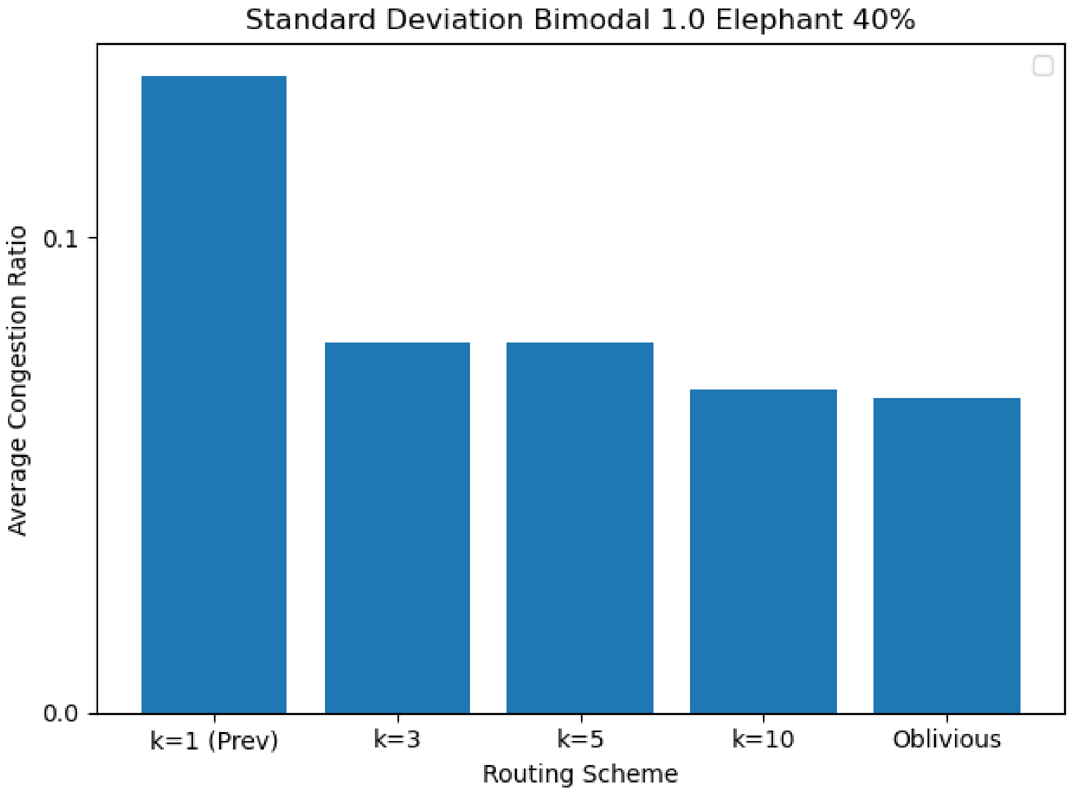
Gravity 0.6
Average Congestion Ratio
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
1.2
1.3
1.4
1.5
1.6
1.7
1.8
k=1 (Prev)
k=3
k=5
k=10
Oblivious
Routing Scheme

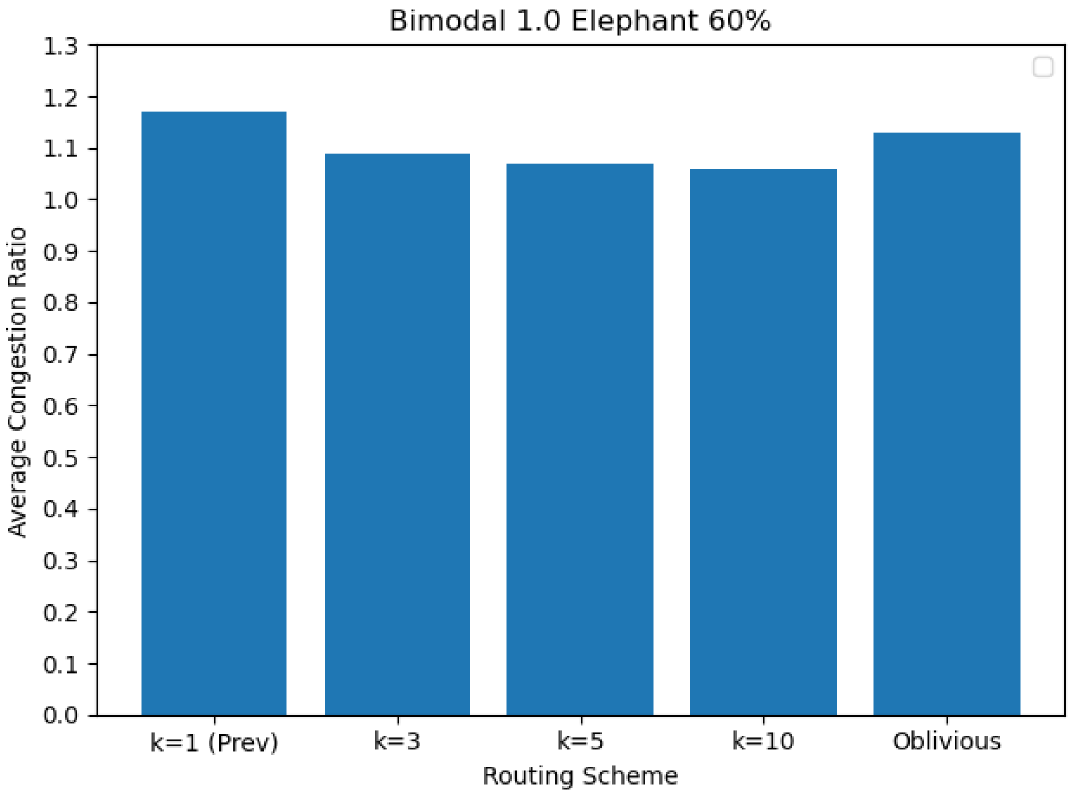

Gravity 0.9
Average Congestion Ratio
1.8
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
k=1 (Prev)
k=3
k=5
k=10
Oblivious
Routing Scheme

Bimodal Traffic:






Standard Deviation Bimodal 1.0 Elephant 40%
Average Congestion Ratio
0.0
0.1
k=1 (Prev)
k=3
k=5
k=10
Oblivious
Routing Scheme


Bimodal 1.0 Elephant 60%
Average Congestion Ratio
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
1.2
1.3
k=1 (Prev)
k=3
k=5
k=10
Oblivious
Routing Scheme

Restoring the Reinforcement Learning Results

The outstanding result of the paper is the prove of concept that a RL agent can be useful to produce good routing scheme that minimize the congestion ratio.

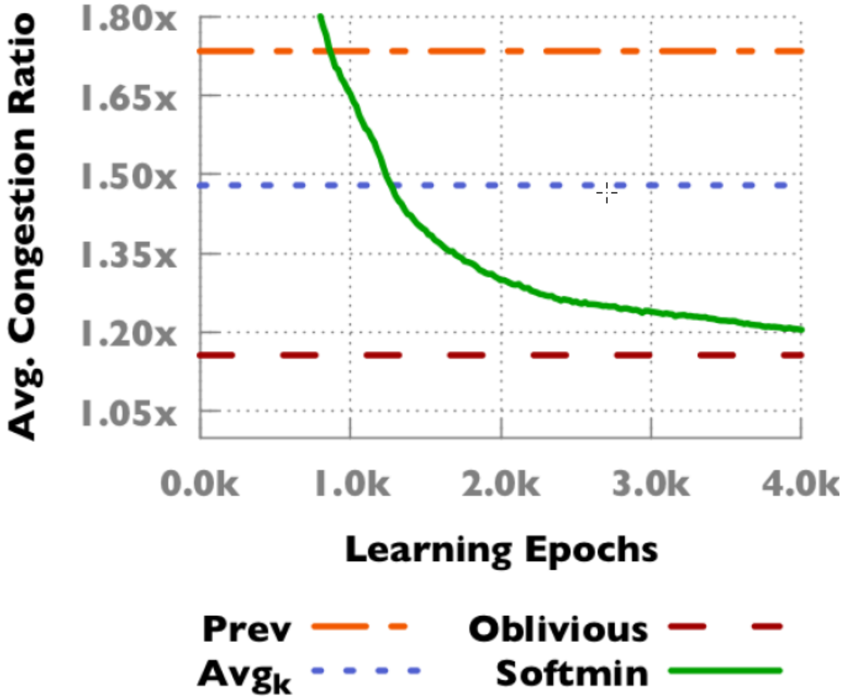
The writers set up an environment that simulate the network nodes and links and every timestep a new traffic matrix with new demands is routed by the agent.

The process of learning is that the agent needs to learn for a map from the traffic matrix to weights for each network link, using that weight the environment calculate shortest path to each destination of demands from every other node. Finally, plug these costs and each edge weight to soft-min function to calculate for each node the percentage of flow carried by each leaving edge. The final step is to run each demand of flow around the network links until all of it reach to its destination and calculate the most congested edge for the final congestion ratio of the traffic matrix.

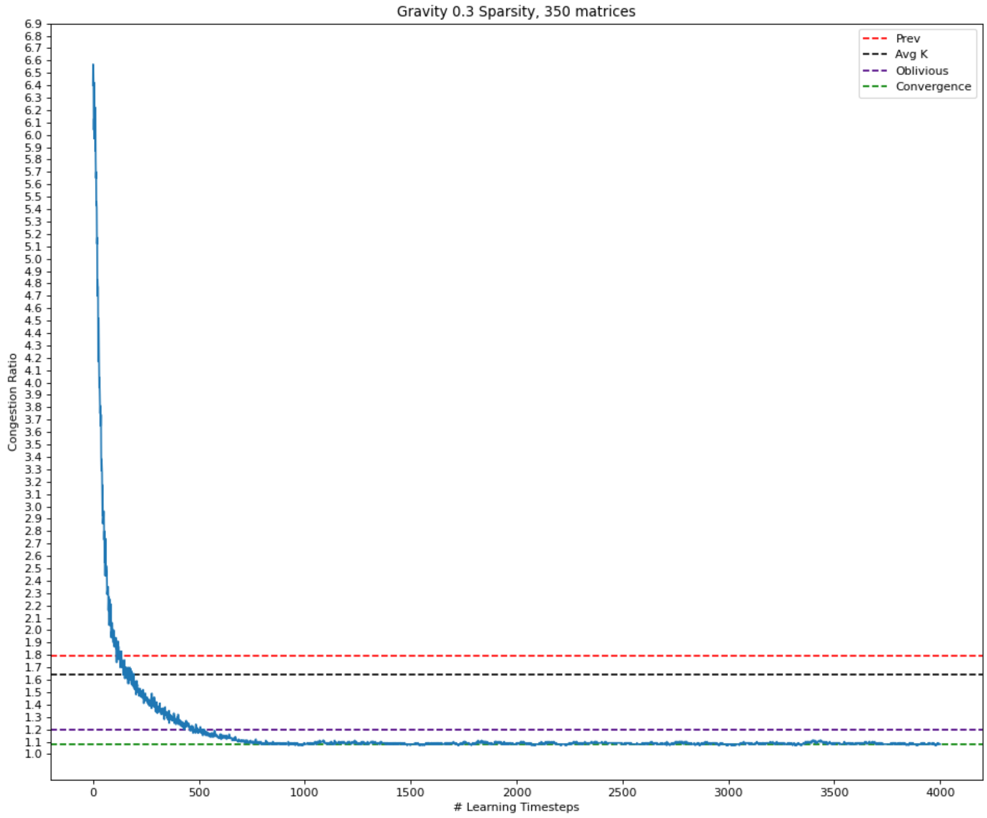
Evaluation

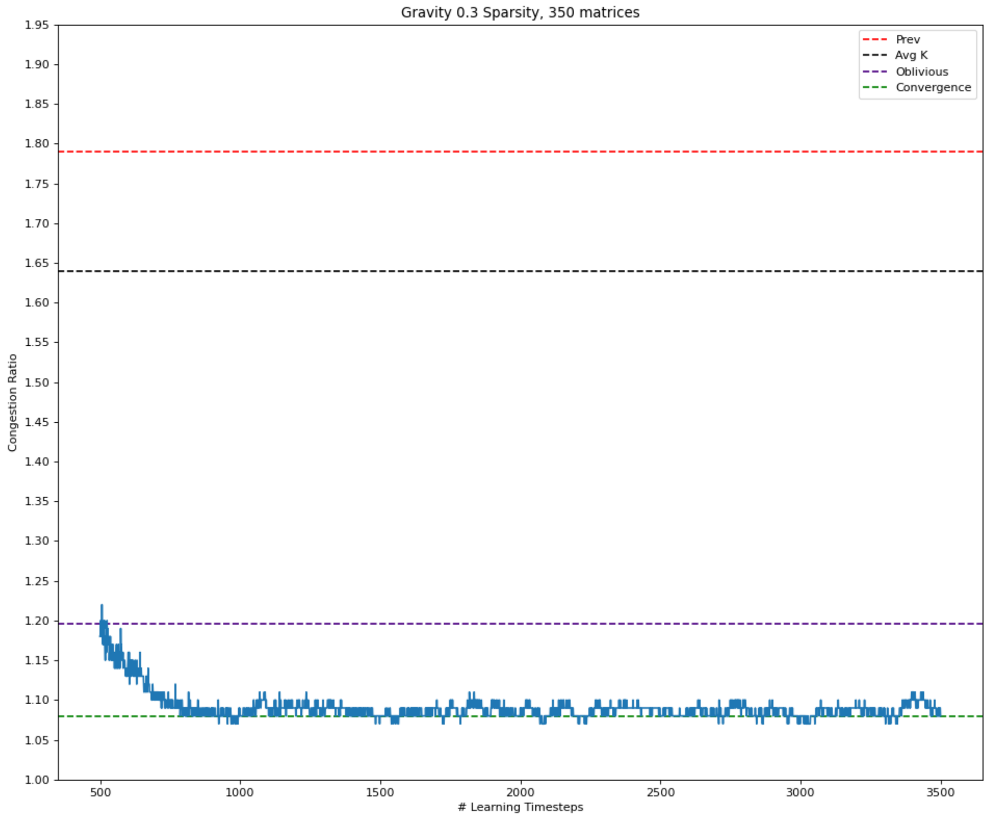
Similar to the paper, a 12-node topology with 26 edges (the original topology includes duplicate edges, this represented as double the capacity for those edges) and constant link capacity of 10,000 Mb.
I evaluate the agent with discount factor equal to 0 and episode length of 1 because flows are not continued more than one timestep, so I assume a myopic approach to minimize the congestion of the current traffic matrix is a good start because there is no relation to future traffic demands.

<u>Gravity Traffic Paper results:</u>

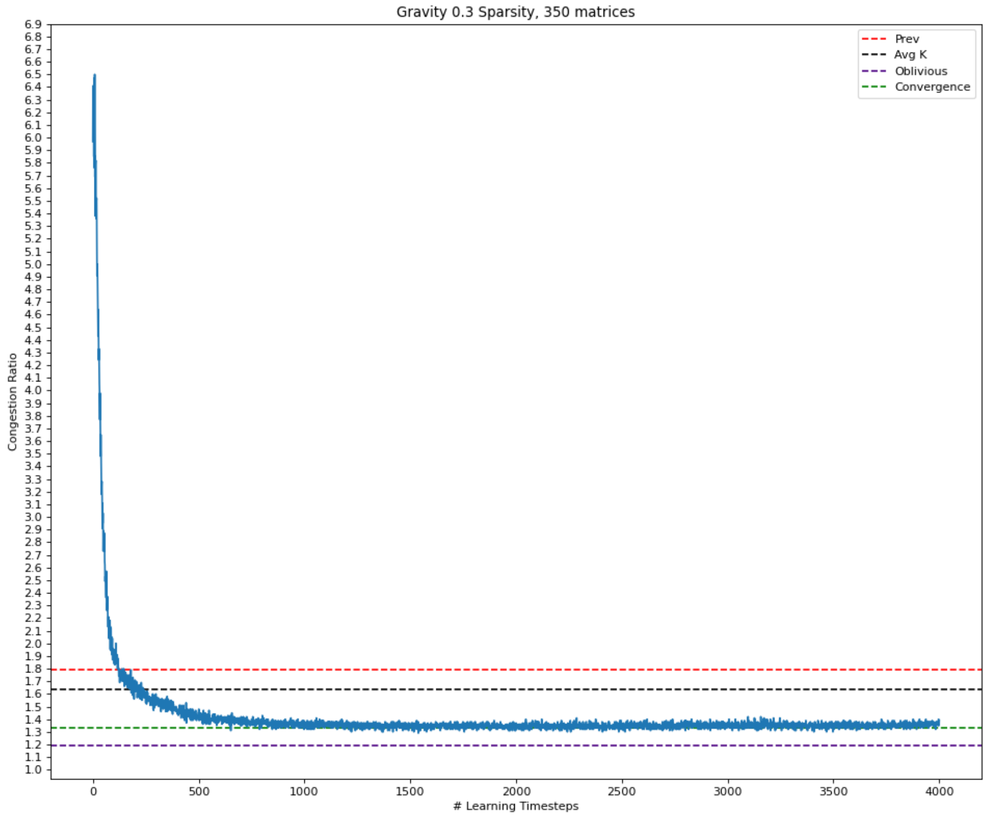


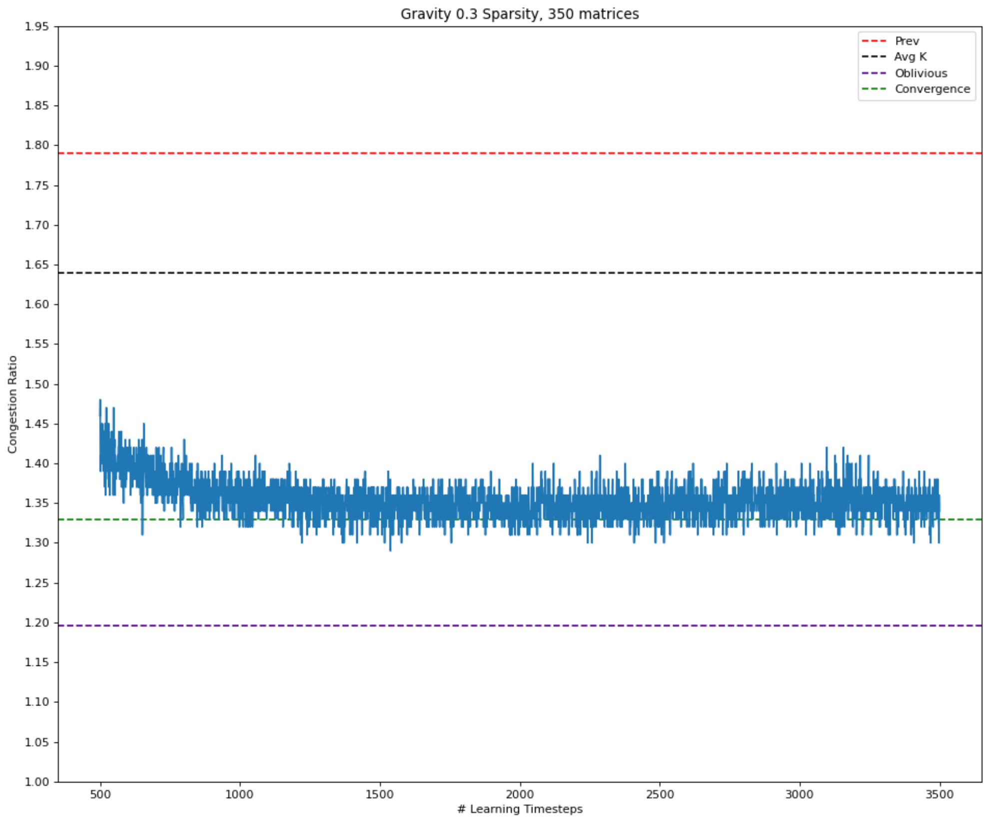
Gravity Traffic with 350 different TMs:



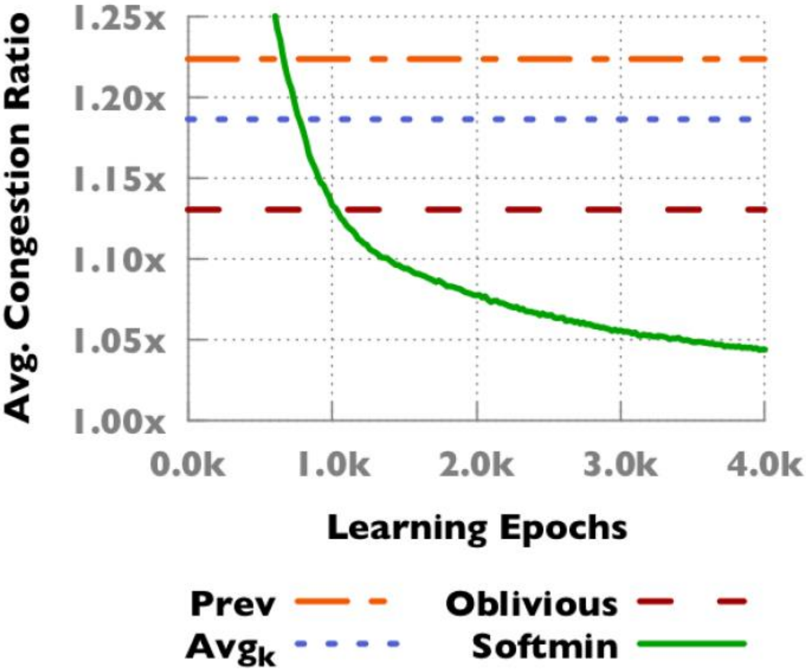


Gravity Traffic with 10,500 different TMs:

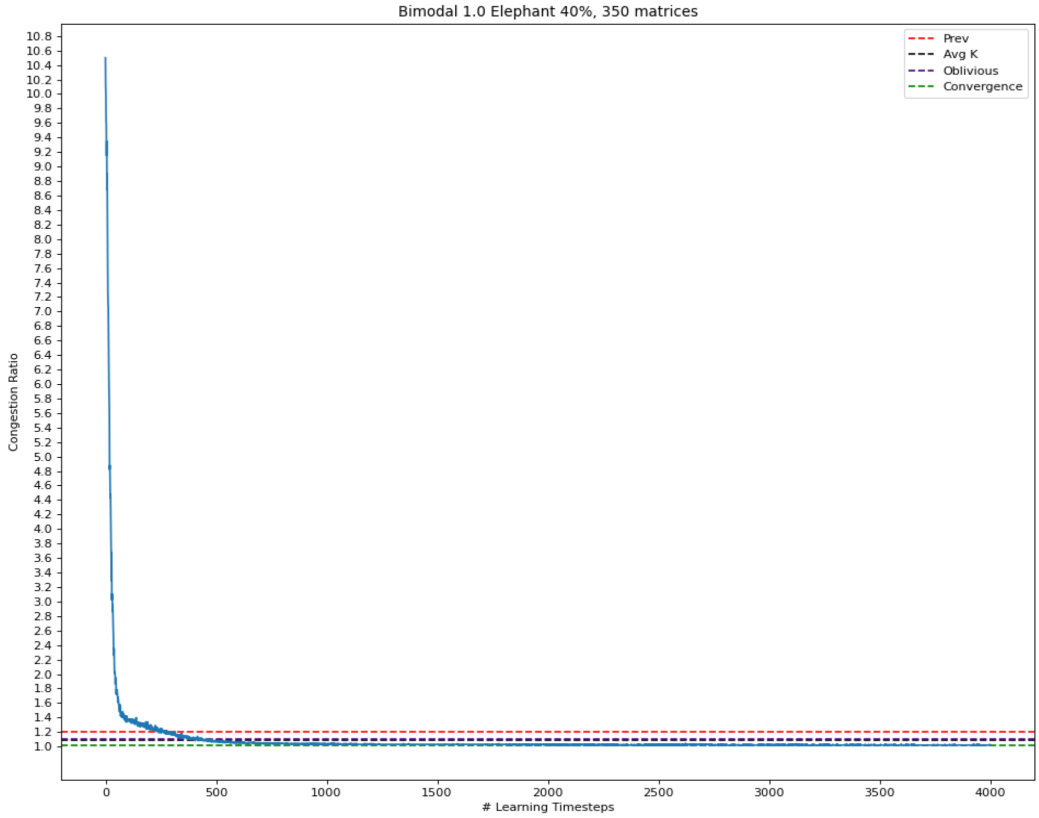


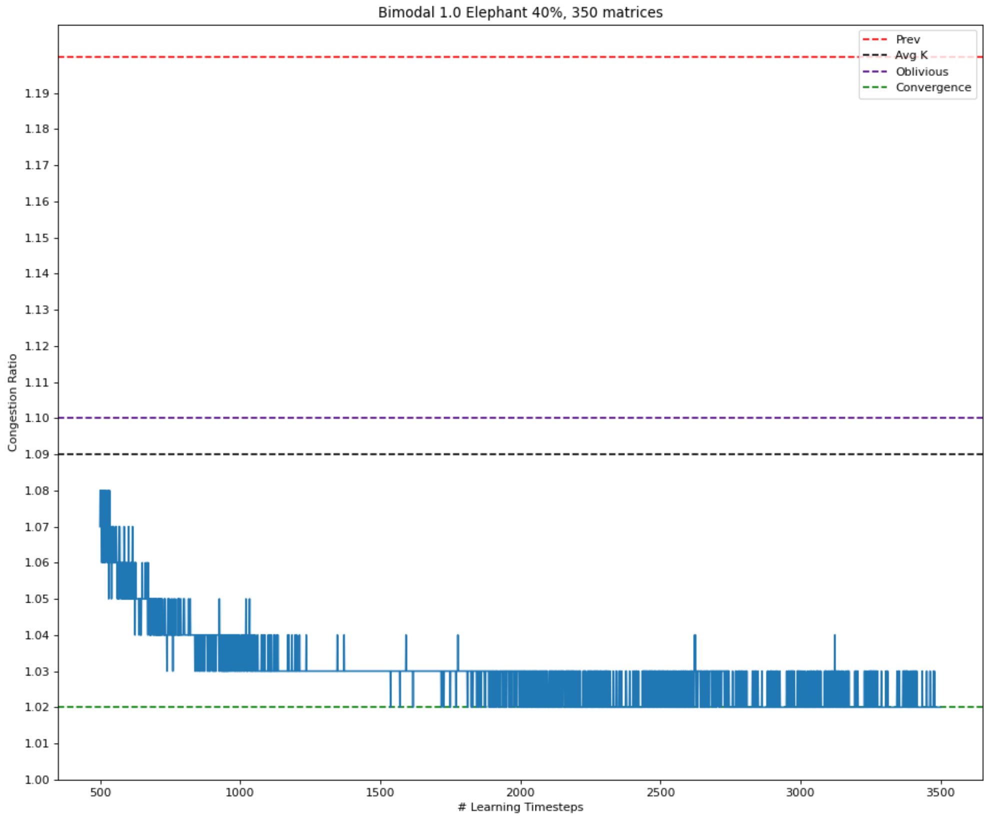


Bimodal Traffic Paper results:

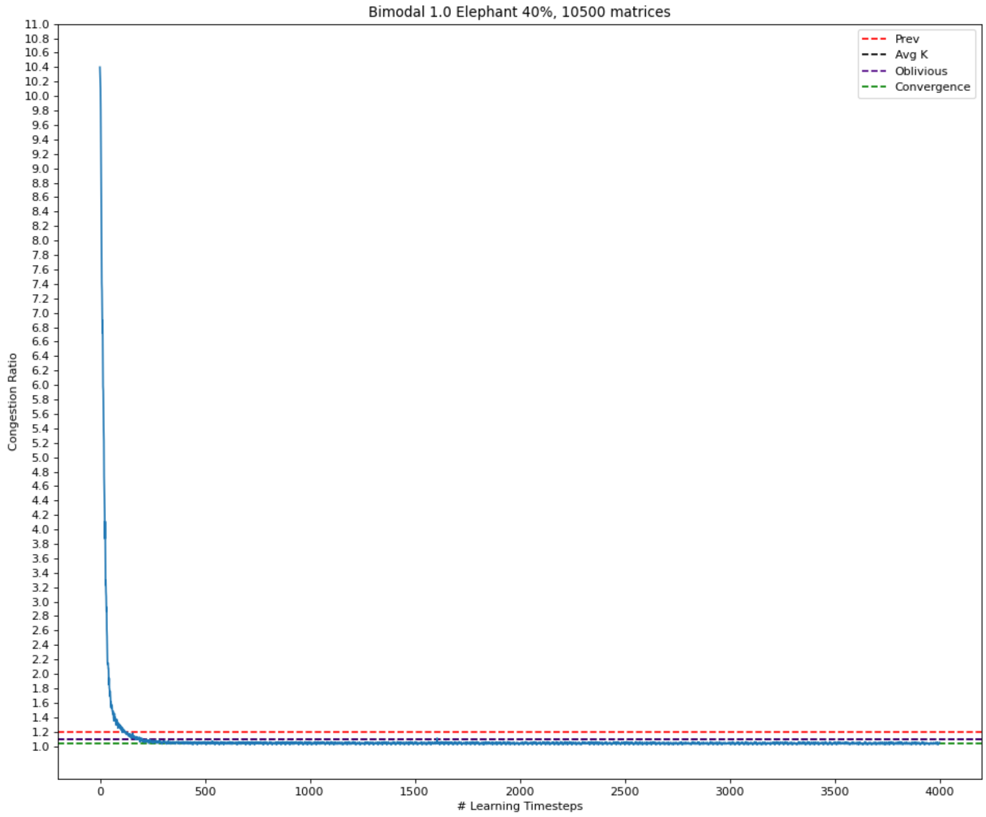


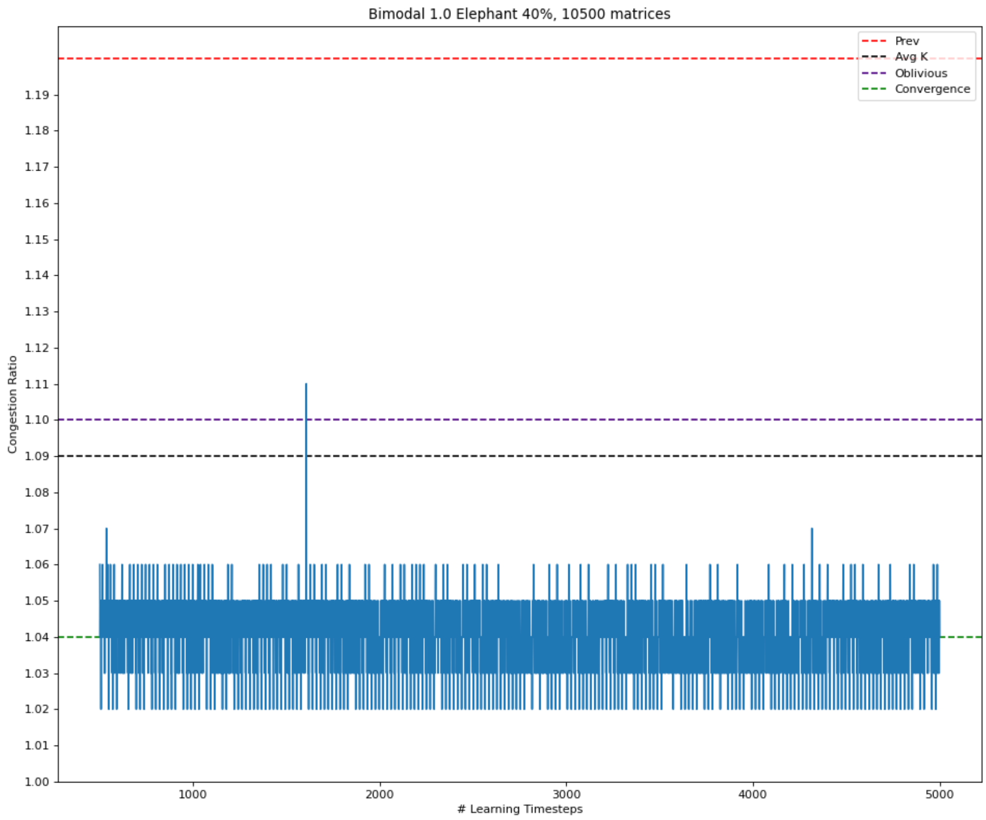
Bimodal Traffic with 350 different TMs:





Bimodal Traffic with 10,500 different TMs:





References

- Valadarsky, A., Schapira, M., Shahaf, D., & Tamar, A. (2017, November). Learning to route. In Proceedings of the 16th ACM workshop on hot topics in networks (pp. 185-191).
- Applegate, D., & Cohen, E. (2006). Making routing robust to changing traffic demands: algorithms and evaluation. IEEE/ACM Transactions on Networking, 14(6), 1193-1206.
- Azar, Y., Cohen, E., Fiat, A., Kaplan, H., & Räcke, H. (2004). Optimal oblivious routing in polynomial time. Journal of Computer and System Sciences, 69(3), 383-394.