GRASP: Accelerating Shortest Path Attacks via Graph Attention

Zohair Shafi[1], Benjamin A. Miller[1], Ayan Chatterjee[1], Tina Eliassi-Rad[1], Rajmonda S. Caceres[2]
[1]Northeastern University, Boston, MA, [2]MIT Lincoln Laboratory, Lexington, MA

Motivation:

Can we use machine learning to speed up Force Path Cut[1], an APX-hard combinatorial optimization problem for attacking shortest paths in a network?

Force Path Cut (FPC) Problem Formulation:

Given a graph $G = (V, E)$, edge weights, edge removal costs, and a path p^* from Source (S) to Target (T) and a budget, can edges be removed with total cost less than the budget such that p^* is the shortest path between S and T?

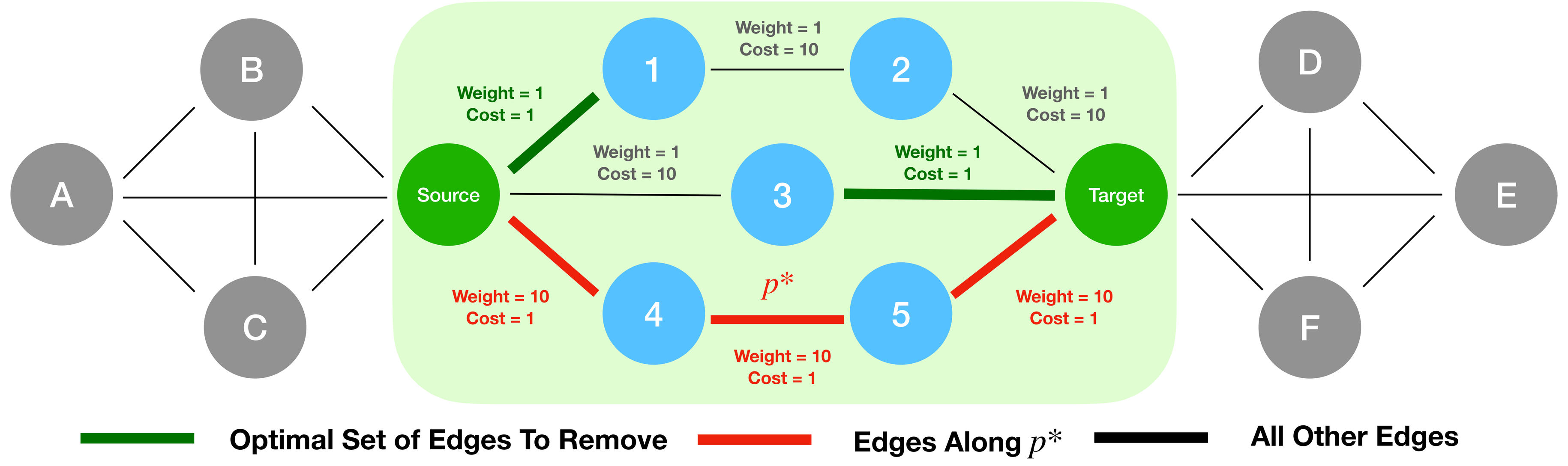

Figure 1: Illustration of the Force Path Cut problem

Our Approach:

- FPC solutions are often contained within a much smaller subgraph of the network (illustrated in Fig. 1)
- Identifying the right subgraph containing FPC solutions can greatly reduce optimization runtime
- We pose the FPC subgraph selection as a learning problem over historical FPC solutions and train a Graph Attention Networks (GAT)[2] to predict the subgraph of interest
- Our algorithm GRASP, iteratively proposes a subgraph to optimize over and calls out an optimization module to produce the solution. In our current implementation we use the PATHATTACK [1] algorithm as the optimization module

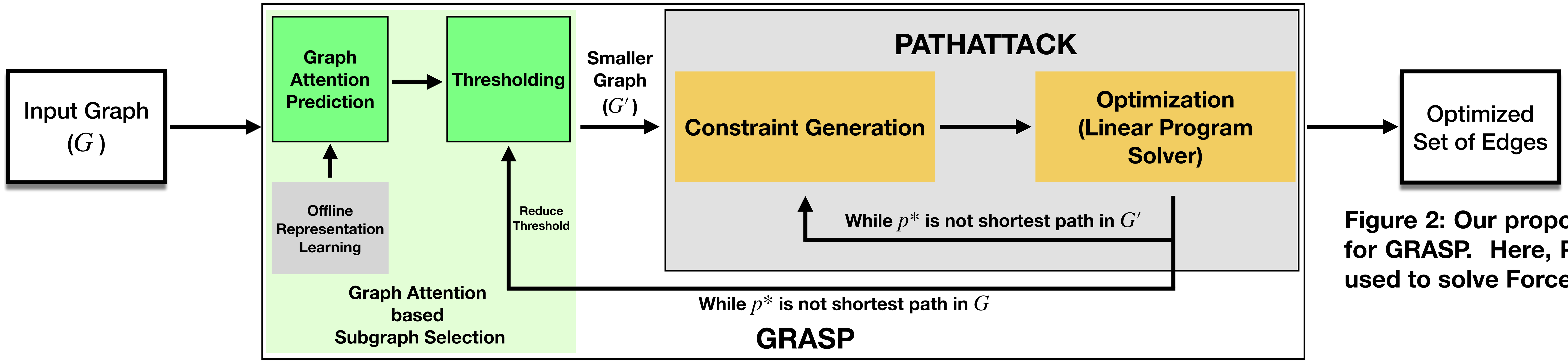

Figure 2: Our proposed architecture for GRASP. Here, PATHATTACK[1] is used to solve Force Path Cut

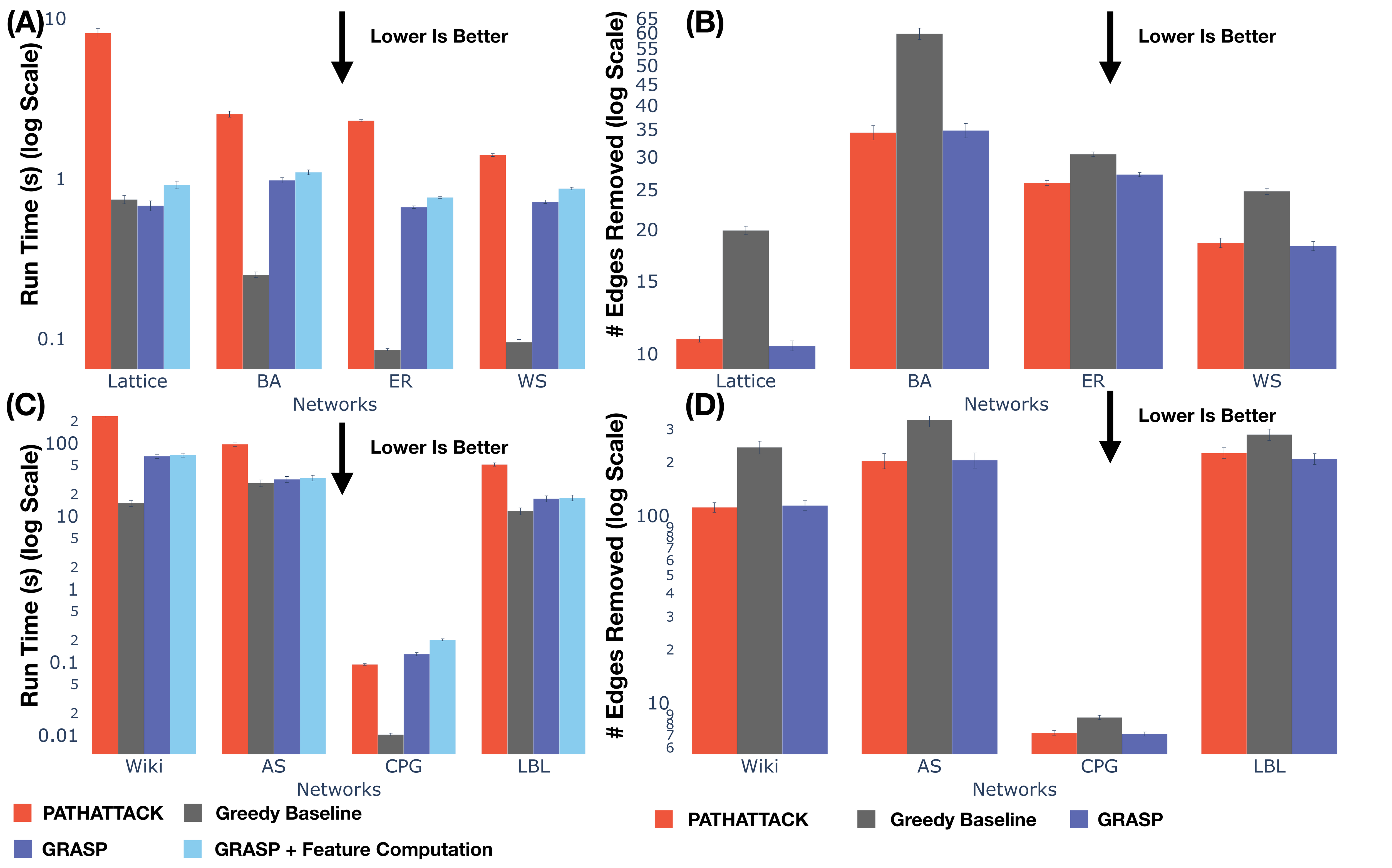

Figure 3: The first row of figures show results on synthetic graphs, with results on the real-world graphs on the second row. Observe shorter runtimes for GRASP when compared to PATHATTACK, while cutting similar number of edges. Note that greedy baseline, while faster, removes a high number of edges leading to lower quality of solution

[1] Miller et al. Attacking Shortest Paths by Cutting Edges. arXiv preprint arXiv:2211.11141, 2022
[2] Veličković et al. Graph attention networks, ICLR 2018

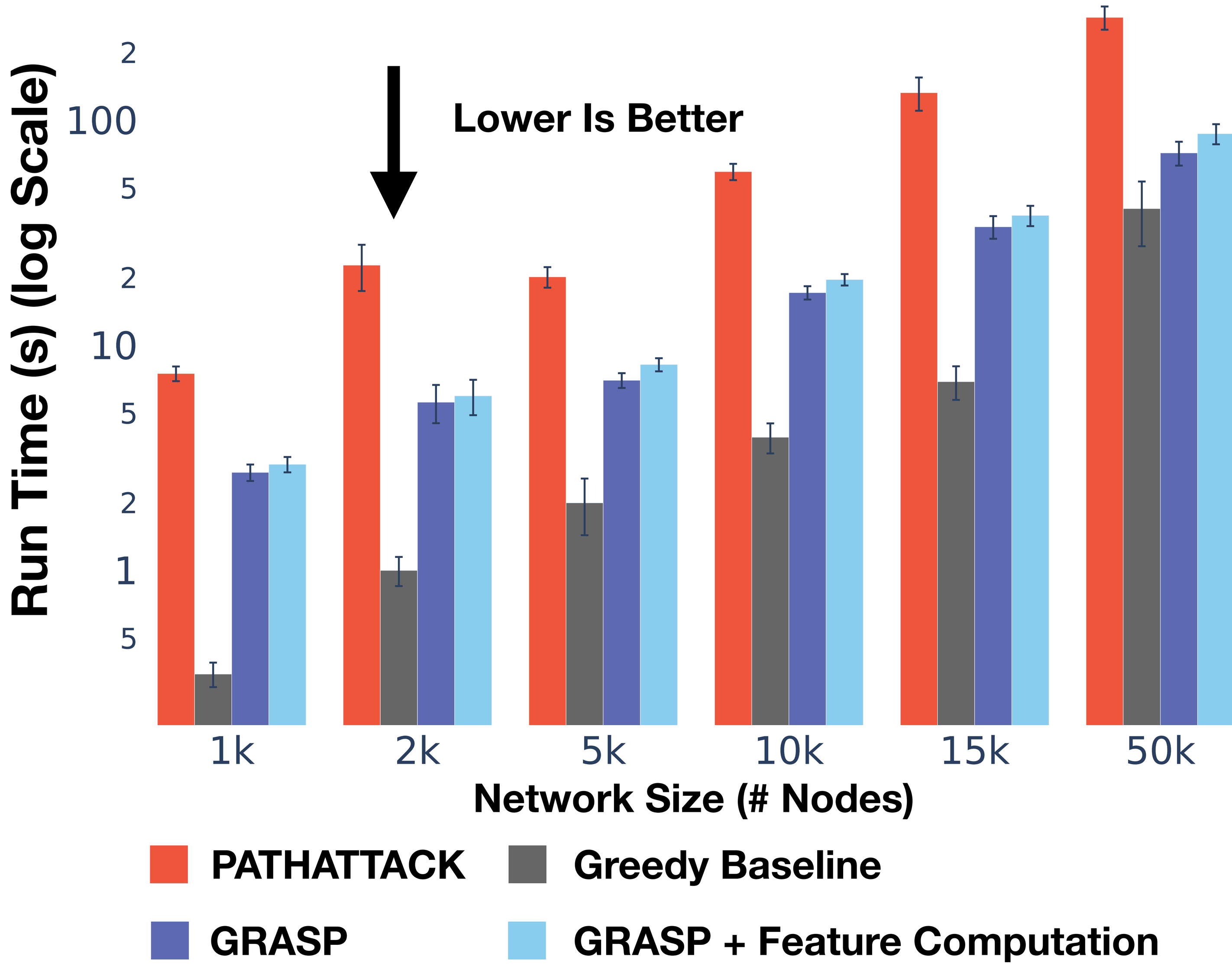

Figure 4: Scalability of GRASP trained on a BA graph with 1000 nodes

Main Contributions:

1. GRASP achieves **faster run times** (~10x in the best case) when compared to the competitive PATHATTACK approach
2. GRASP retains **solution quality** - a challenge for greedy, fast baseline
3. GRASP **generalizes well** to different network topologies and network sizes

Distribution Statement A. Approved for public release. Distribution is unlimited. This material is based upon work supported by the Under Secretary of Defense for Research and Engineering under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Under Secretary of Defense for Research and Engineering.