Maximum Network Flow

Muhammad Shahrom Ali | Kabir Kumar | Fatima Nadeem | Mazeyar Moeini Feizabadi

Crux of the Problem

How to get the maximum flow in a flow network

Flow

Network

Flow Network

Maximum Flow

Not Solvable with Quantum Computers

Next: Definitions

Network

- A Directed Graph
- Each **node**: an actor
- Each **edge**: a connection between two actors
- Residual Graph: A graph of residual flows

$$c_f(e) = c(e) - f(e)$$

Flow

- The amount of substance being carried.
- For example:
  - Data packets in a computer network
  - Goods on a trade-route/network
  - Water in a network of pipes
- Mathematically:

  A Real Function

Next: More Definitions

Flow Network

- A weighted Network
- Each **weight**: the capacity of that connection
- **Source:** node with in-degree = 0
- **Sink:** node with out-degree = 0
- The Flow assigns a real value that must be ≤ capacity

Maximum Flow

- The maximum amount of quantity that can flow through the network

Next: Theorem

Max-flow Min-cut theorem

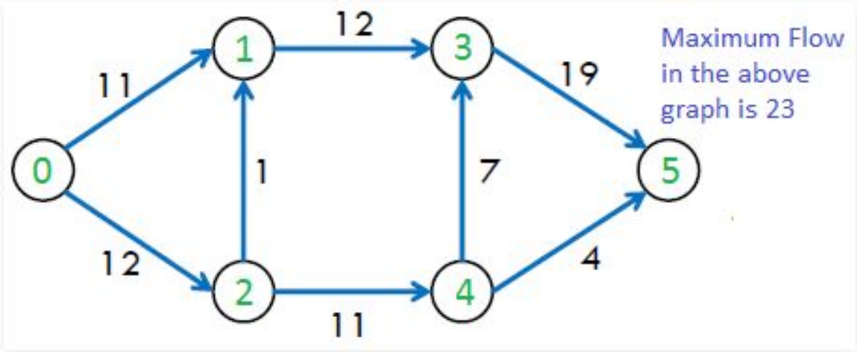
“The maximum flow through any network from a given source to a given sink is exactly the sum of the edge weights that, if removed, would totally disconnect the source from the sink.” [2]

Algorithms Implemented

1. Ford-Fulkerson Algorithm
2. Edmonds-Karp Algorithm
3. Dinic's Algorithm
4. Push-Relabel Algorithm

Next:
Ford-Fulkerson

Ford-Fulkerson Algorithm



- The first attempt towards solving the maximum flow problem
- Can be applied to multiple sources and sinks as well.
- Complexity = O(fE)
  - F = maximum flow
  - E = Edges
- Guarantees correct answer on termination, no guarantee for termination, however.

Ford–Fulkerson

FORD-FULKERSON (G, s, t, c)

FOREACH edge $e \in E : f(e) \leftarrow 0$.

$G_f \leftarrow$ residual graph.

WHILE (there exists an augmenting path P in G_f)

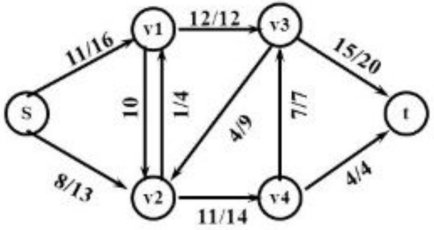
$f \leftarrow$ AUGMENT (f, c, P).

Update G_f.

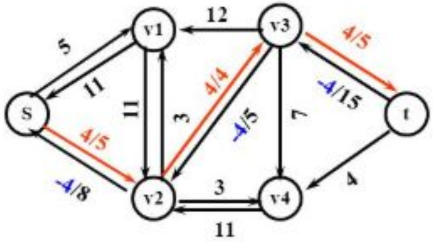
RETURN f.

}

Flow network $G = (V,E)$

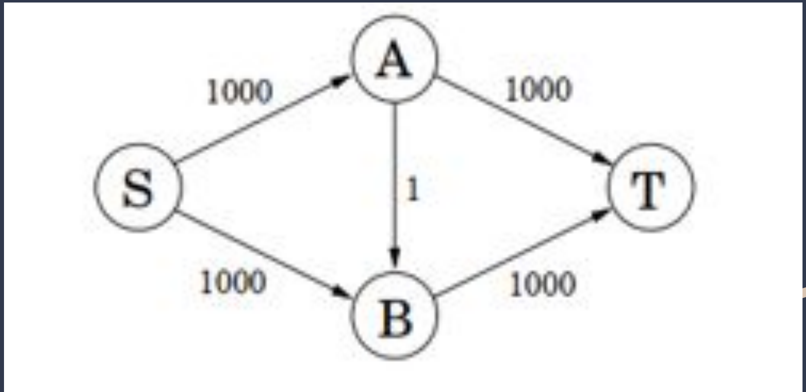


residual network $G_f = (V,E_f)$



Our virtual flow f_p along the augmenting path p in G_f

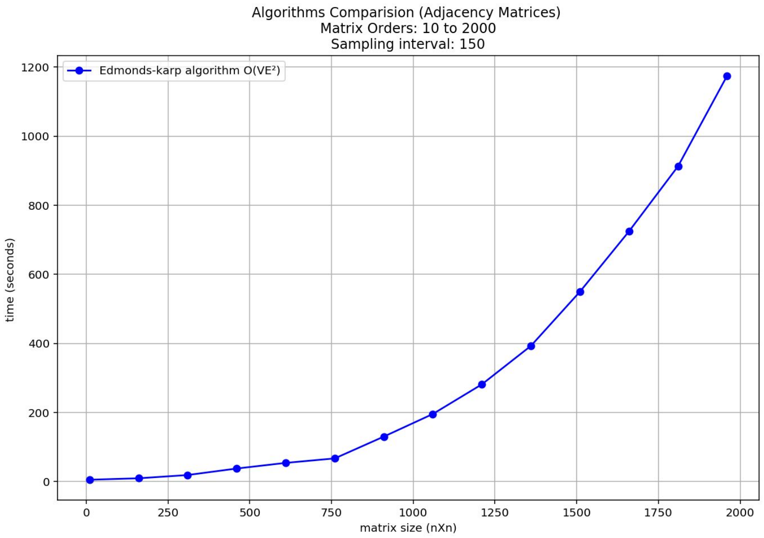
Ford-Fulkerson Algorithm (cont.)



- This example is an eye opening one.
- Careless updates can lead to around a 1000 iterations.
- There are cases when it can run on without terminating, although involving irrational capacities. A generalized non terminating case is discussed by Uri Zwick in his paper. (7)

Next: Edmonds-Karp

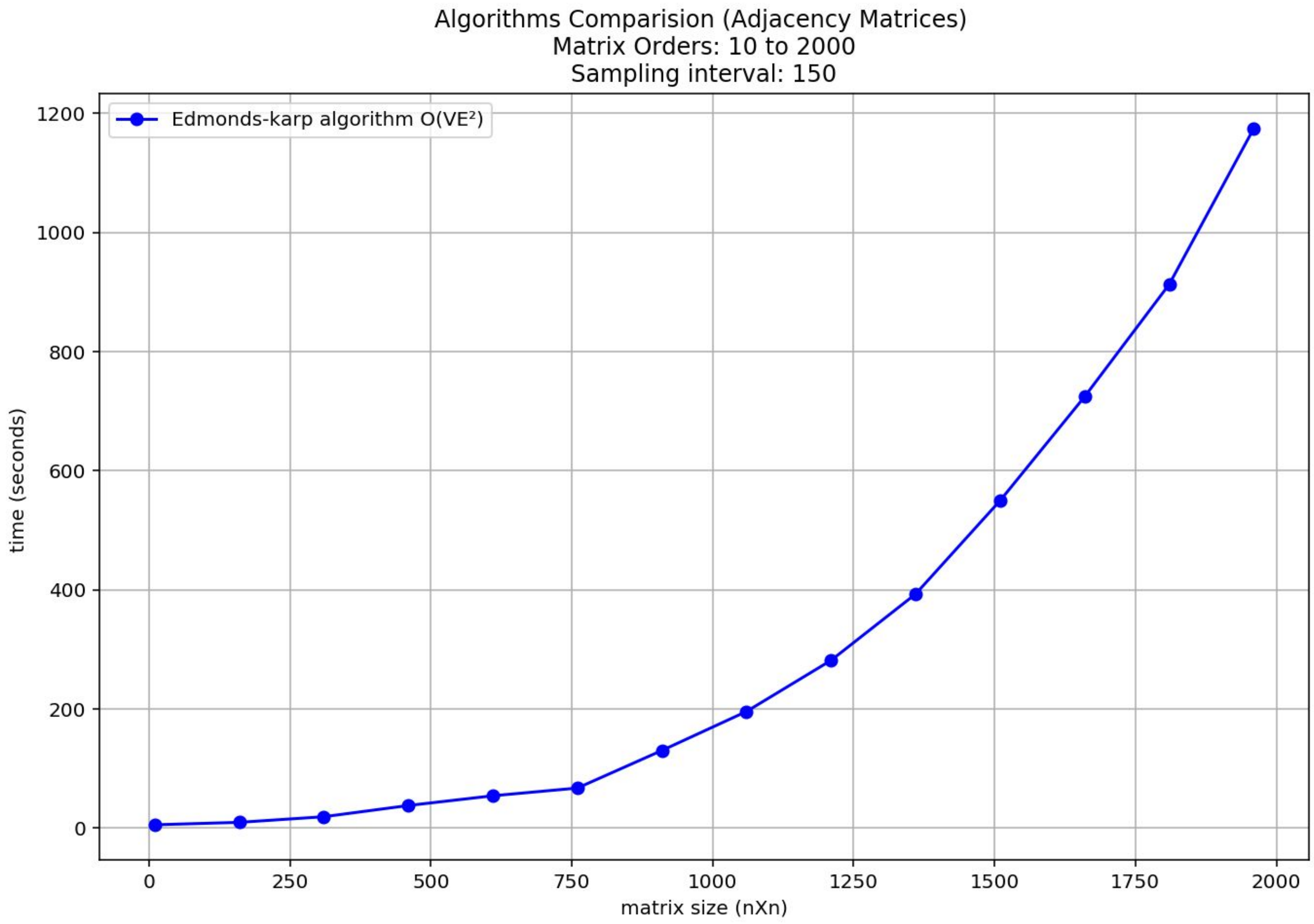
Edmonds-Karp Algorithm



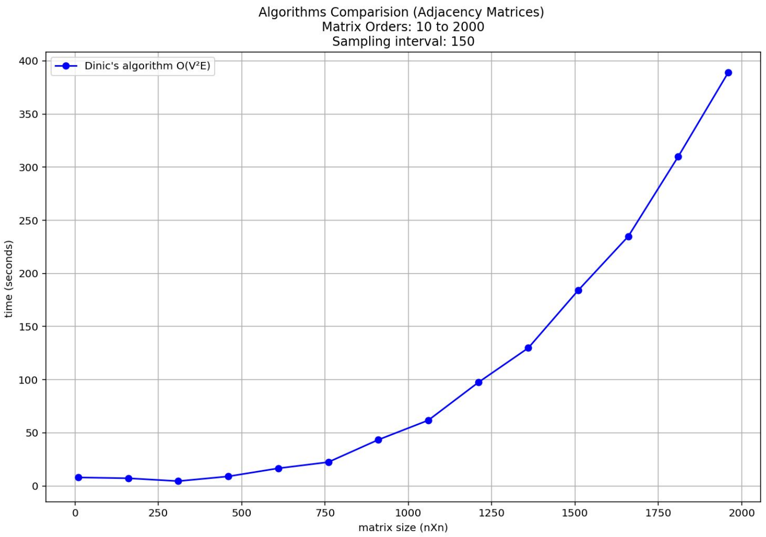
- A more defined version of Ford-Fulkerson Algorithm
- Must use Breadth-First Search to look for Augmenting Paths
- Augmenting paths are simply any path from the source to the sink that can currently take more flow. [1]
- $O(|V|.|E|^2)$

1. Brilliant.org

Next:
Dinic's Algorithm


Algorithms Comparision (Adjacency Matrices)
Matrix Orders: 10 to 2000
Sampling interval: 150
Edmonds-karp algorithm O(VE²)
time (seconds)
matrix size (nXn)
0
200
400
600
800
1000
1200
0
250
500
750
1000
1250
1500
1750
2000

Dinic’s Algorithm

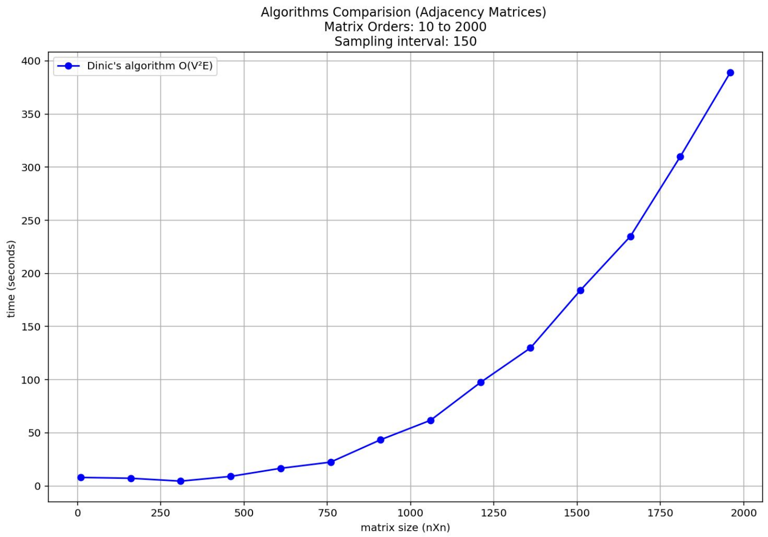


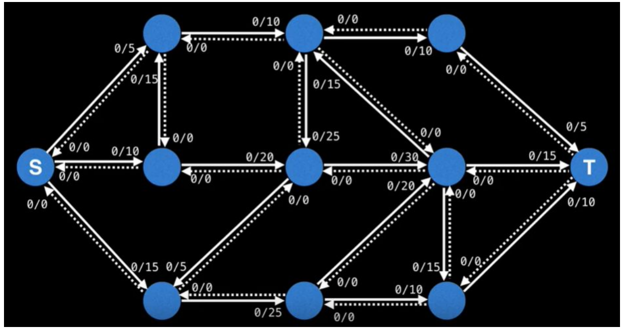
- Strongly **polynomial algorithm** invented by Computer scientist **Yefim Dinitz** published in **1970**

- Time complexity = **O((V^2)E)**

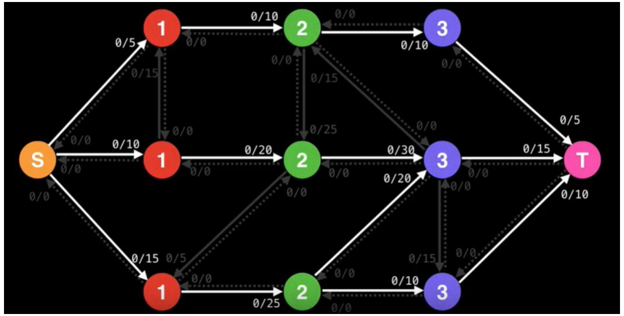
Next:
Push-Relabel

Dinic's Algorithm



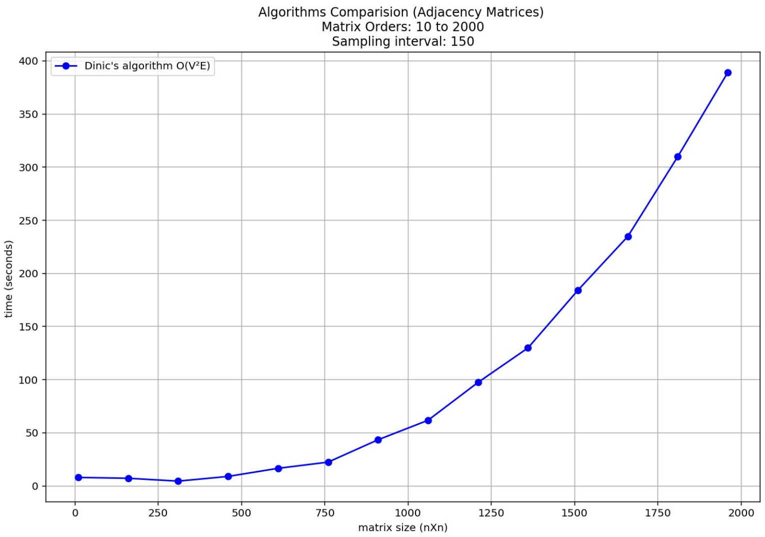


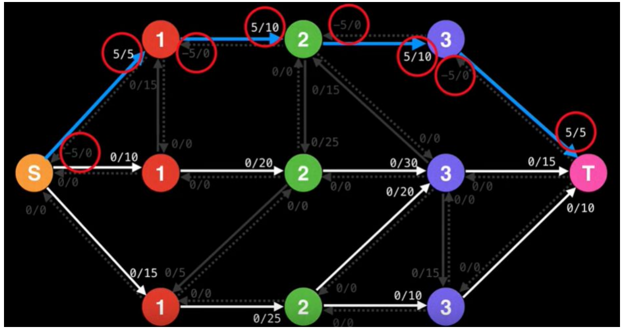
- Runs **BFS** from the source to form **level graph (O(E))**.
- Only main edges and residual edges with **positive progress** and **remaining capacity (capacity - flow) > 0** are considered.



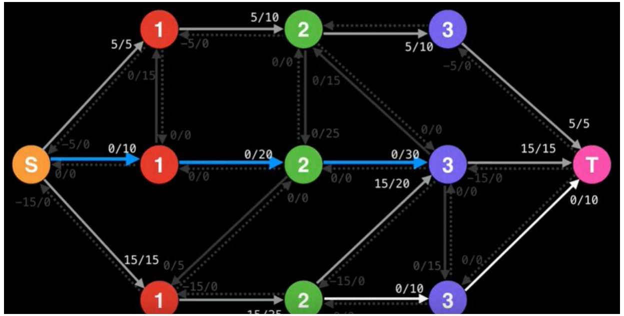
Next:
Push-Relabel

Dinic’s Algorithm



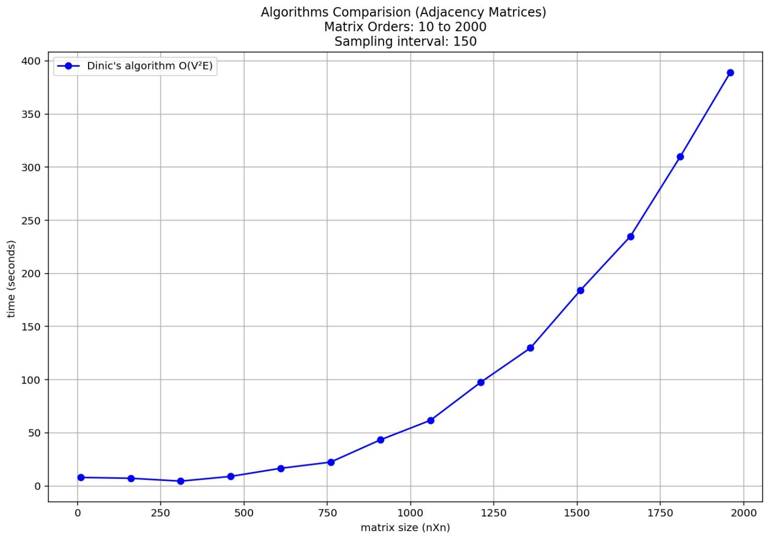


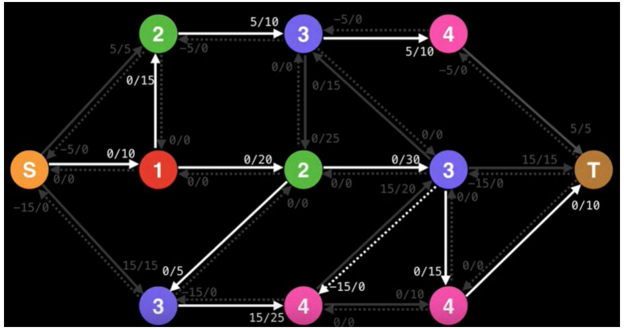
- Runs **DFS** from source to sink, multiple times, to find all possible **augmenting paths** and their **bottleneck values** until **blocking flow** is reached **(O(VE))**.



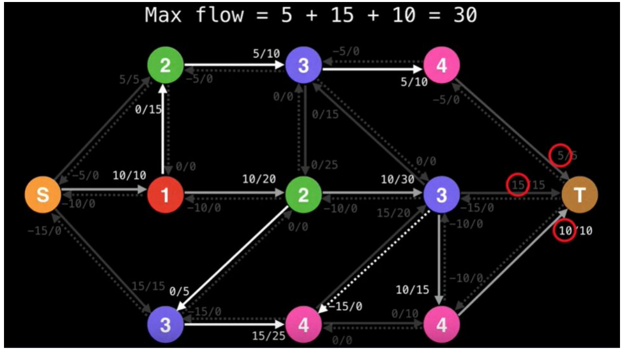
Next:
Push-Relabel

Dinic’s Algorithm

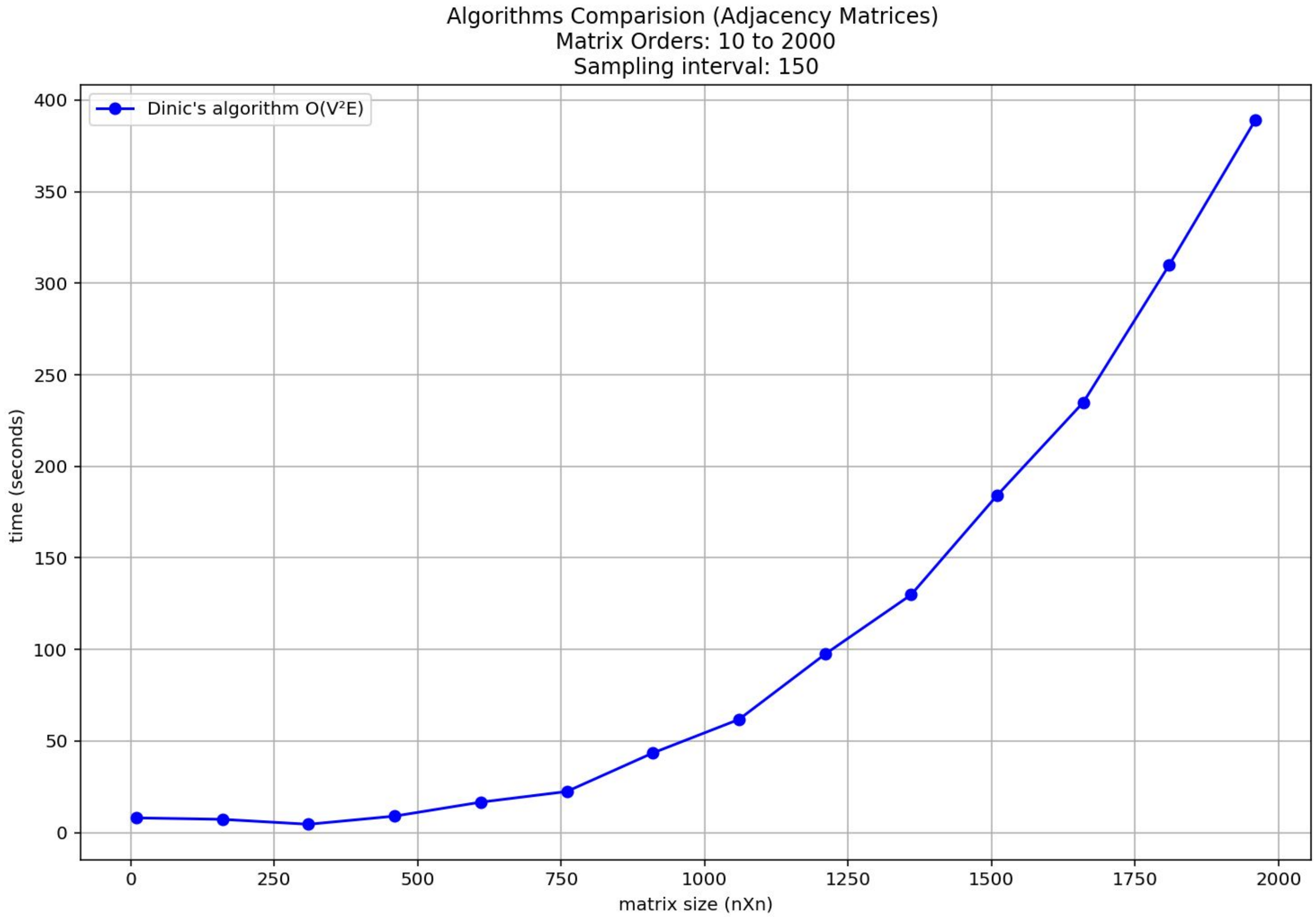




- **Repeat** the process **(O(V))** until the graph is completely saturated; **no more augmenting paths** from source to sink left.
- **Maximum flow = sum of all bottleneck values**

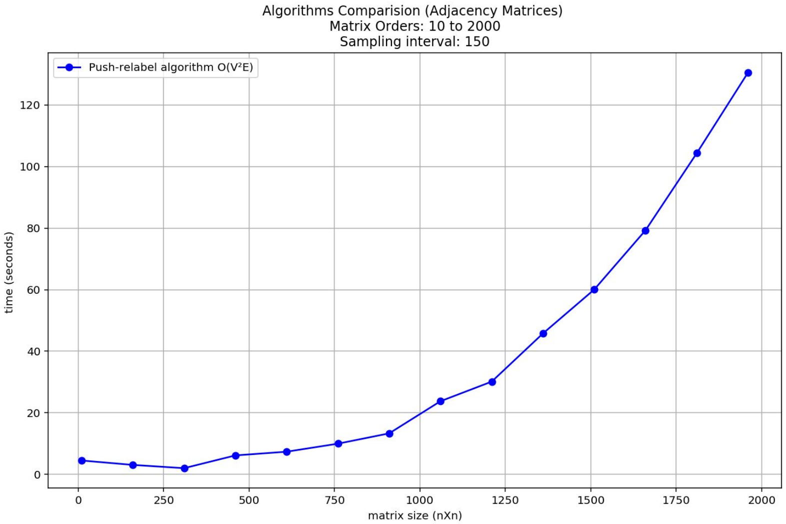


Next:
Push-Relabel



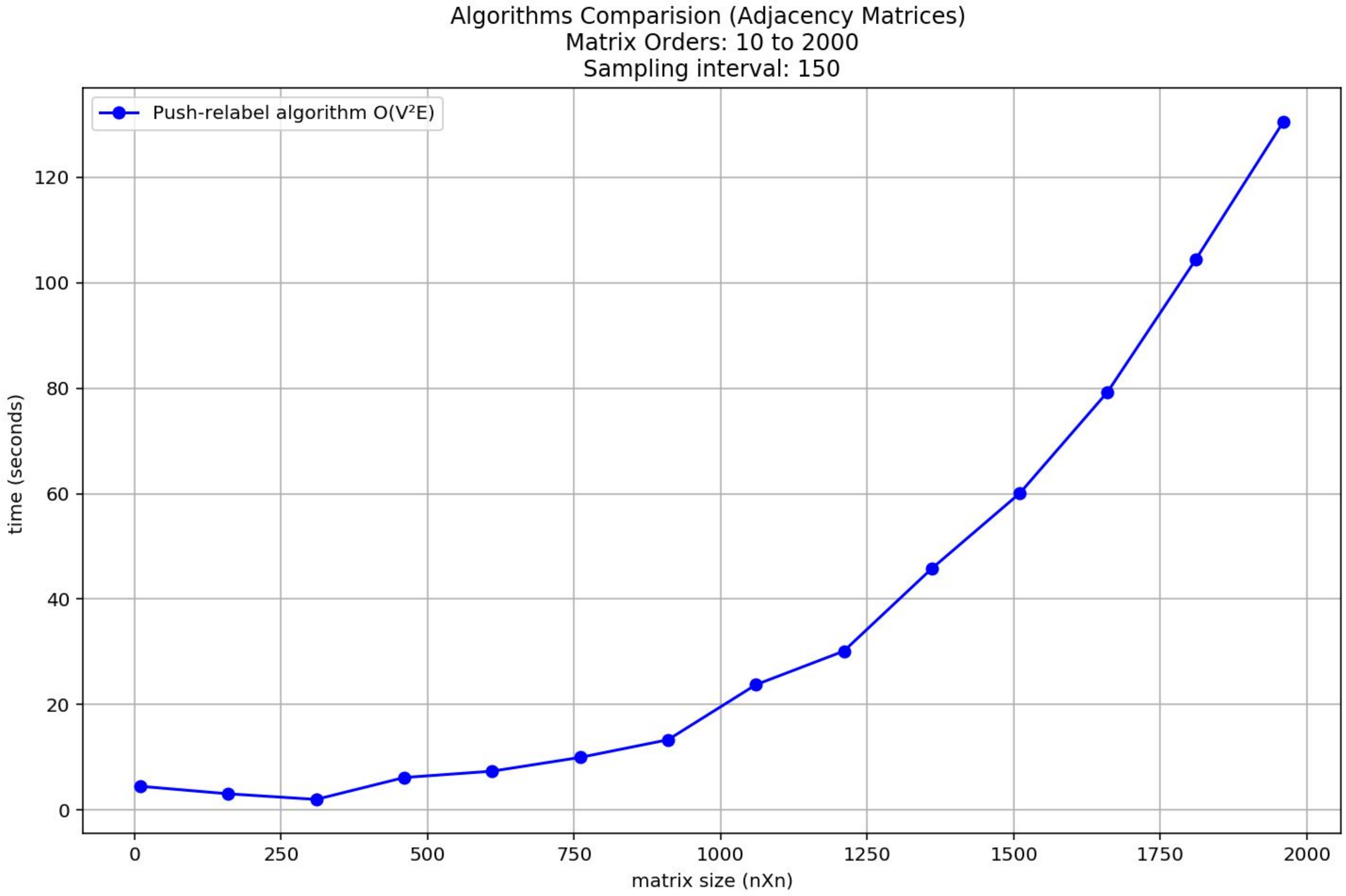
Algorithms Comparision (Adjacency Matrices)
Matrix Orders: 10 to 2000
Sampling interval: 150
Dinic's algorithm O(V²E)
time (seconds)
matrix size (nXn)
0
50
100
150
200
250
300
350
400
0
250
500
750
1000
1250
1500
1750
2000

Push-Relabel Algorithm



- Compared to the Ford-Fulkerson algorithm Push-Relabel works of local operations known as.
- We imagine the vertices as being pipe junctions and and the edges as being pipes.
- We also maintain a list of the height of each vertex, vertex (u->v) iff u.height > v.height
- The push operation is pushing a graphs excess flow to its neighbor with valid height. Note: junctions don't have a limit
- The relabel operation is relabing the height of a junctions.
- We keep doing these operations until can not be done anymore.
- This algorithm has been proven correct.

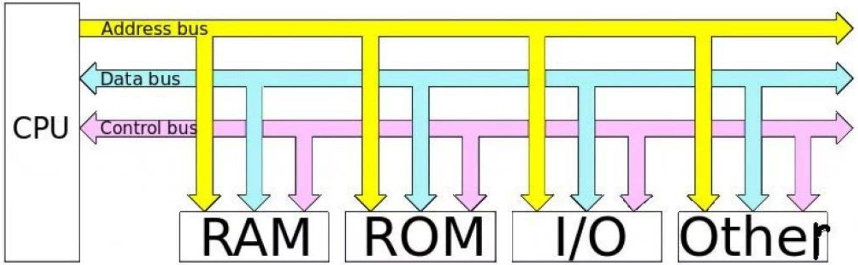
Next: Applications

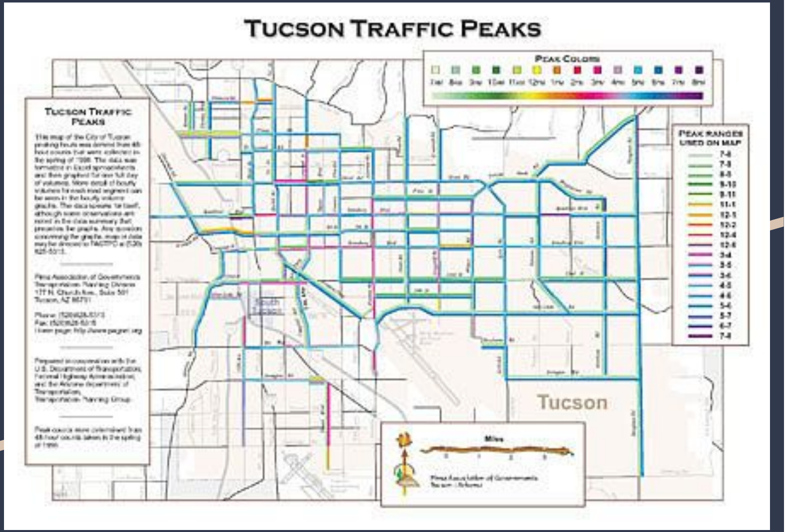


Algorithms Comparision (Adjacency Matrices)
Matrix Orders: 10 to 2000
Sampling interval: 150
Push-relabel algorithm O(V²E)
time (seconds)
0
20
40
60
80
100
120
0
250
500
750
1000
1250
1500
1750
2000
matrix size (nXn)

Applications

MotherBoard Bus ->





<- City Traffic Grid

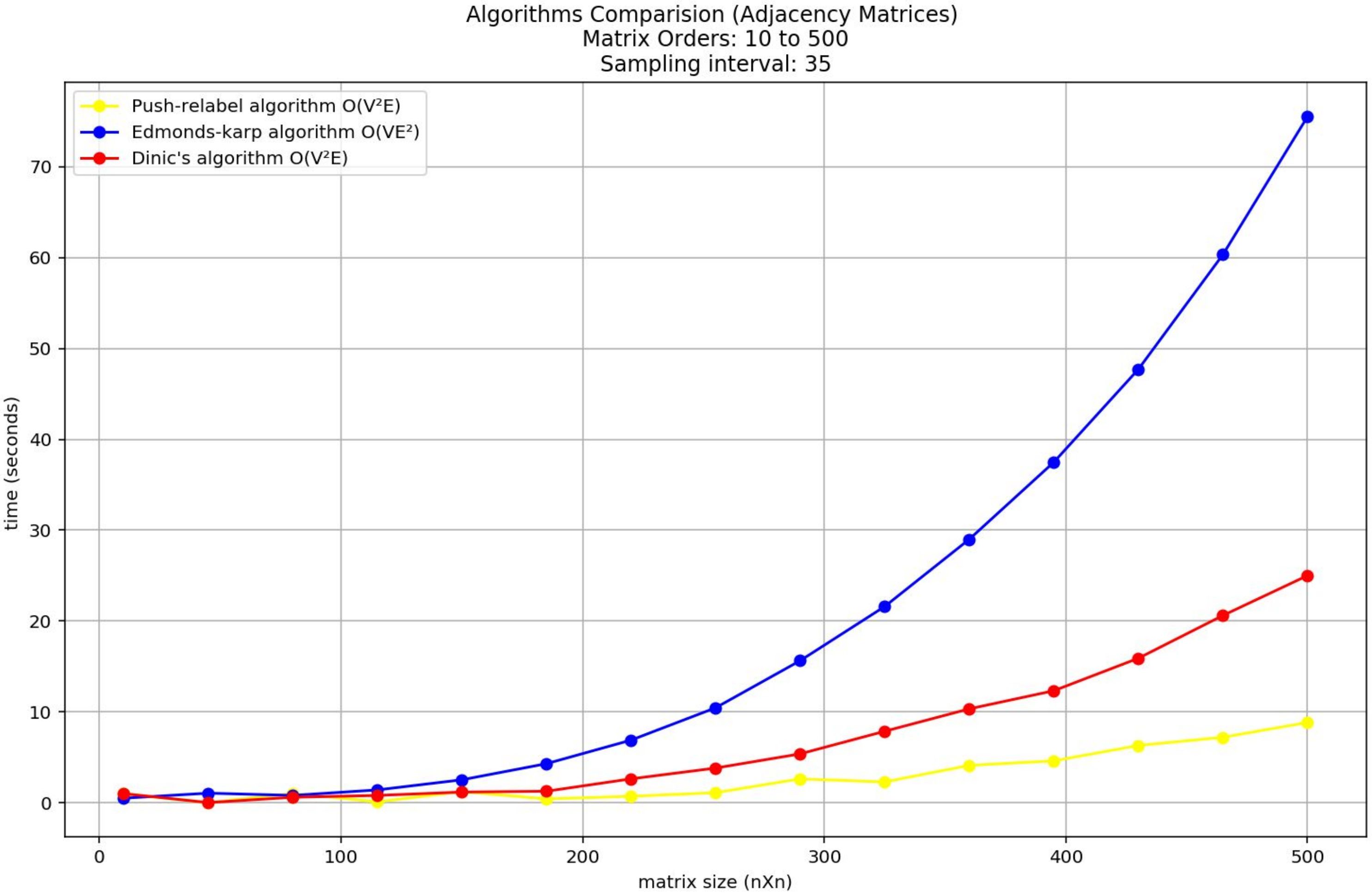
Next: Comparison

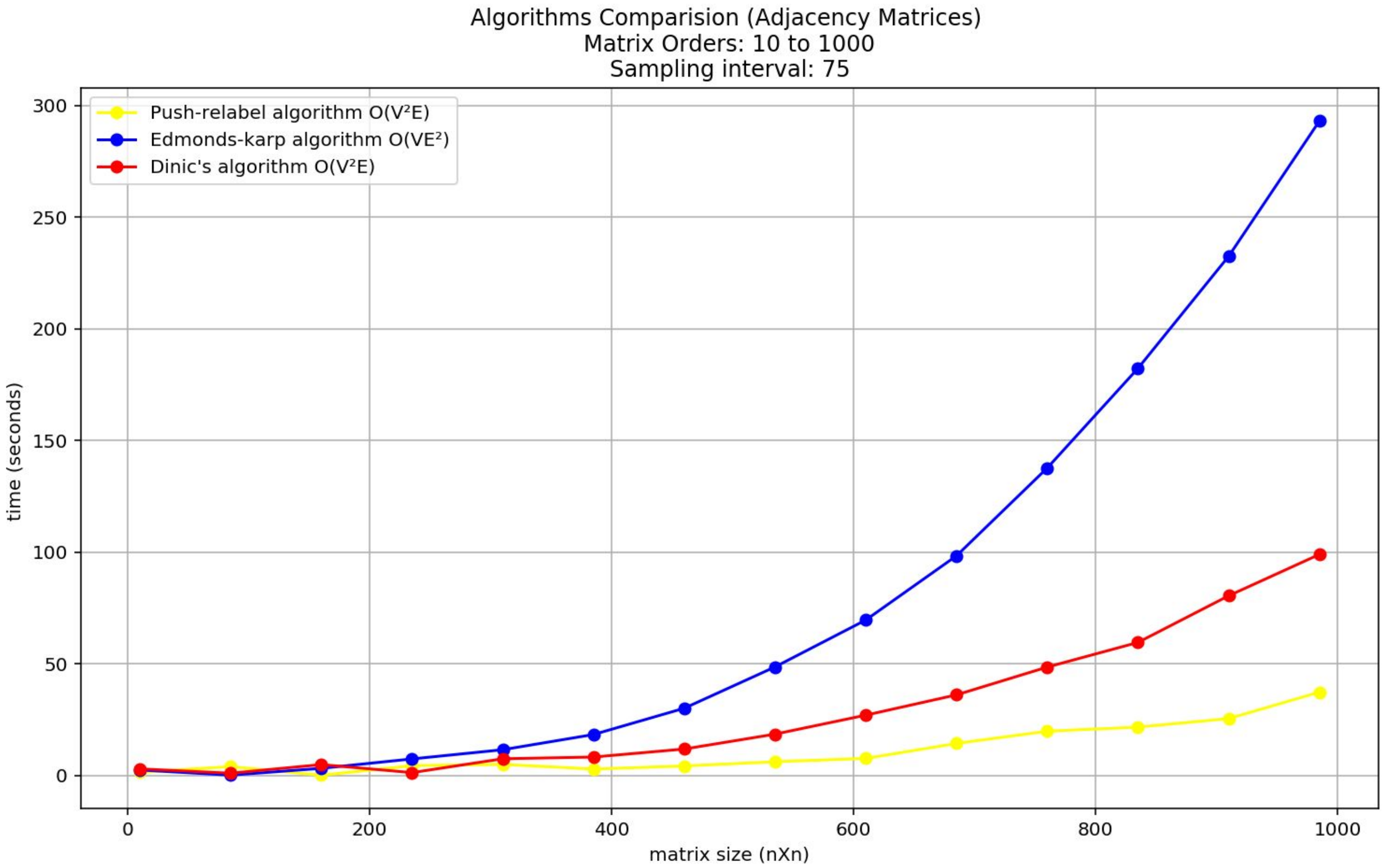
Experiment

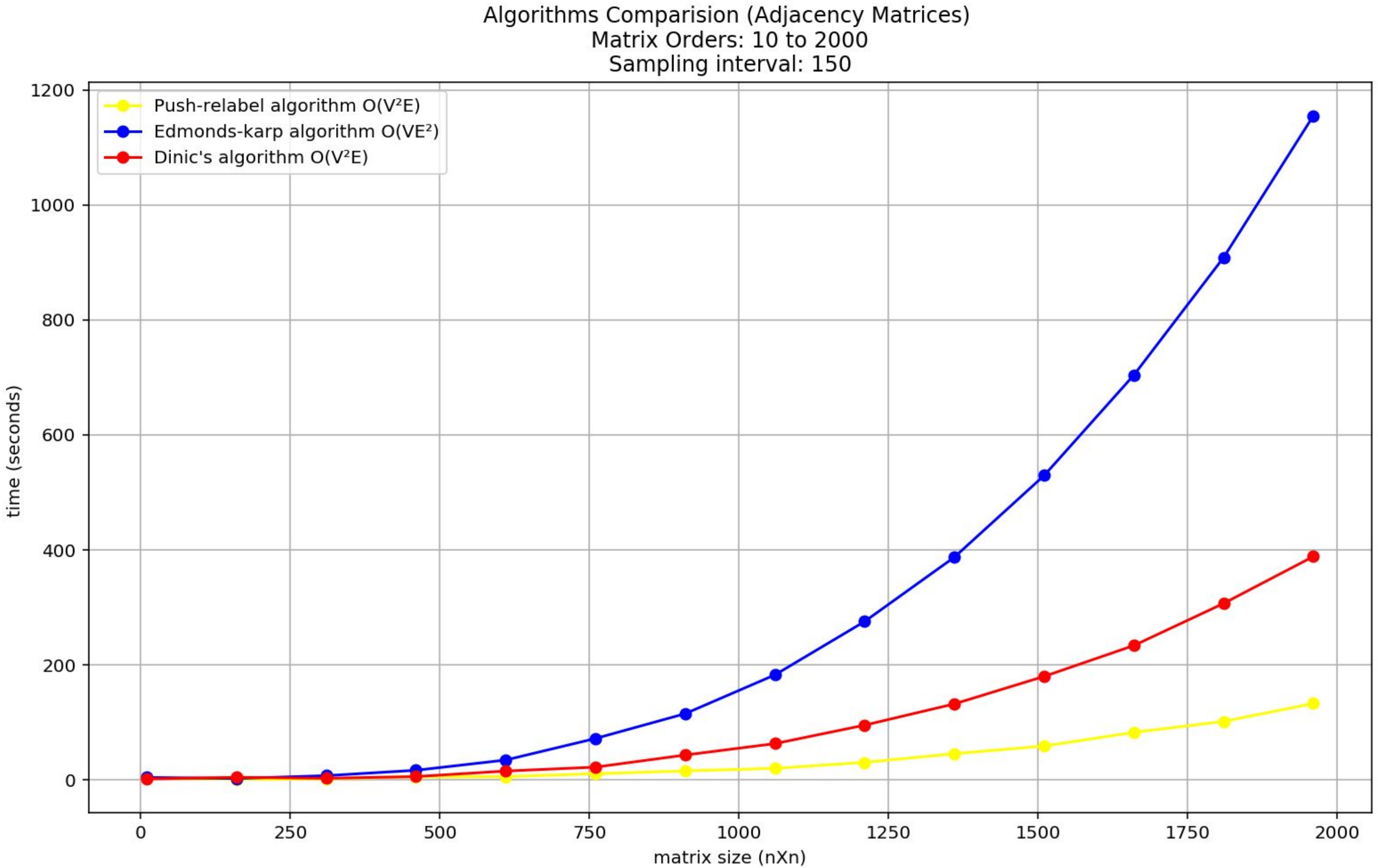
```
def GenerateNetwork(lengthG):
    #Generate adjacency (Capacity) matrix of order lengthG wit
a sink
    random.seed(100)
    G = [[0 for i in range(lengthG)] for j in range(lengthG)]
    for i in range(lengthG):
        for j in range(len(G[i])):
            capacity = random.randint(-20, 15)
            if capacity > 0 and j != i:
                if G[j][i] == 0:
                    G[i][j] = capacity
    return G
```

-The graphs were non-dense adjacency matrices (run time was prioritised over memory)
-Step size was random
-There were some anomalous points in the graph

Machine was a 14nm Processor with 3.6GHz clock and 8GB of RAM


Algorithms Comparision (Adjacency Matrices)
Matrix Orders: 10 to 500
Sampling interval: 35
Push-relabel algorithm O(V²E)
Edmonds-karp algorithm O(VE²)
Dinic's algorithm O(V²E)
time (seconds)
matrix size (nXn)
0
10
20
30
40
50
60
70
100
200
300
400
500


Algorithms Comparision (Adjacency Matrices)
Matrix Orders: 10 to 1000
Sampling interval: 75
Push-relabel algorithm O(V²E)
Edmonds-karp algorithm O(VE²)
Dinic's algorithm O(V²E)
time (seconds)
matrix size (nXn)
0
50
100
150
200
250
300
0
200
400
600
800
1000


Algorithms Comparision (Adjacency Matrices)
Matrix Orders: 10 to 2000
Sampling interval: 150
Push-relabel algorithm O(V²E)
Edmonds-karp algorithm O(VE²)
Dinic's algorithm O(V²E)
time (seconds)
matrix size (nXn)
0
200
400
600
800
1000
1200
0
250
500
750
1000
1250
1500
1750
2000

Citations

1. https://www.geeksforgeeks.org/push-relabel-algorithm-set-1-introduction-and-illustration/
2. https://brilliant.org/wiki/edmonds-karp-algorithm/
3. https://graphonline.ru/en/
4. https://www.cse.unt.edu/~tarau/teaching/AnAlgo/Edmonds%E2%80%93Karp%20algorithm.pdf
5. https://cp-algorithms.com/graph/edmonds_karp.html
6. https://www.cs.cmu.edu/~ckingsf/bioinfo-lectures/netflow.pdf
7. Zwick, U. (1995). *The smallest networks on which the Ford-Fulkerson maximum flow procedure may fail to terminate. Theoretical Computer Science, 148(1), 165–170.* doi:10.1016/0304-3975(95)00022-0
8. https://www.geeksforgeeks.org/ford-fulkerson-algorithm-for-maximum-flow-problem/
9. https://www.cs.princeton.edu/courses/archive/spring13/cos423/lectures/07NetworkFlowI.pdf
10. https://www.youtube.com/watch?v=M6cm8UeeziI - William Fiset - Dinic's Algorithm | Network Flow | Graph Theory

That's about it.