Image Segmentation

Drew Guarnera, Bharath Bogadamidi,
Heather Michaud

The Algorithms

Primary Components of our Algorithm

- Breadth First Search (BFS)
- Ford-Fulkerson (FF)
- Threshold and Penalty Calculations

Breadth First Search

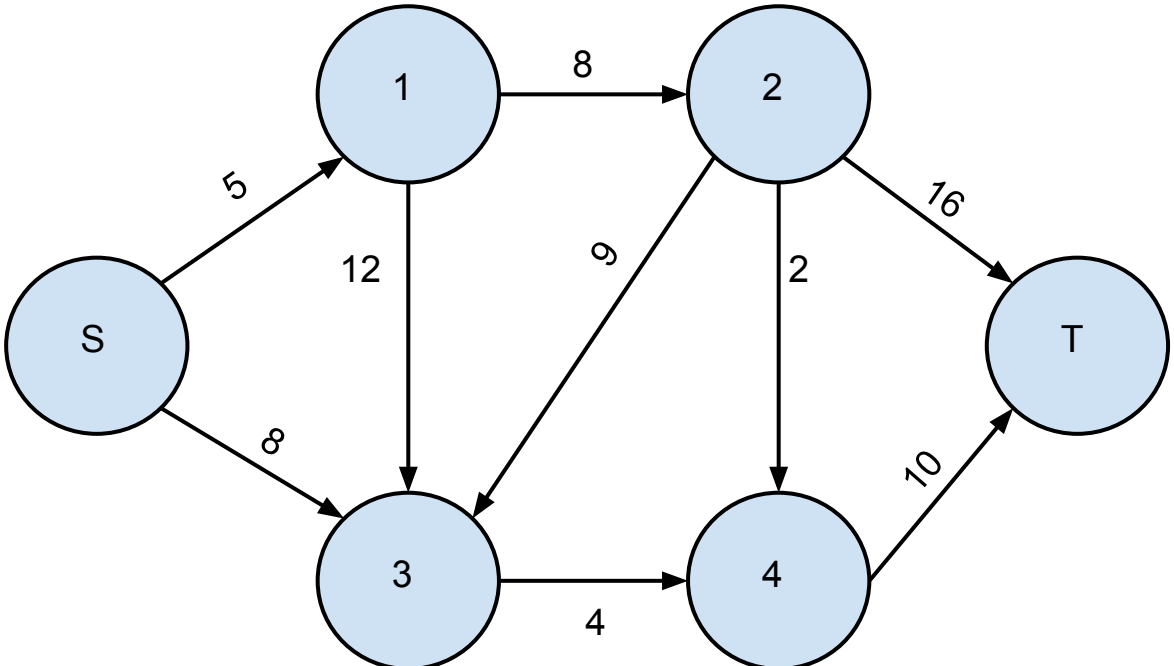
- Mostly Standard
  - Queue to hold node for processing
  - Non-recursive to avoid stack overhead
- Optimizations
  - Arrays to track neighbors/visited and edge weights
  - Early loop termination when end node is found
  - Returns max flow of path (smallest capacity/bottleneck)
- Purpose
  - BFS will help the Ford-Fulkerson algorithm to find optimal paths and provide the flow value used in augmenting the residual graph

Ford-Fulkerson

- Mostly Standard:
  - Edge augmentation
- Optimizations:
  - Use BFS to find paths (Edmond-Karp)
  - Map<int, Map<int, Vertex>> used for adjacency list keeps traversals to a minimum and makes residual graph manipulation straightforward
- Purpose:
  - Supply a residual graph to indicate what pixels are in the foreground (connected to source) and background (connected to sink)

Combined Algorithm

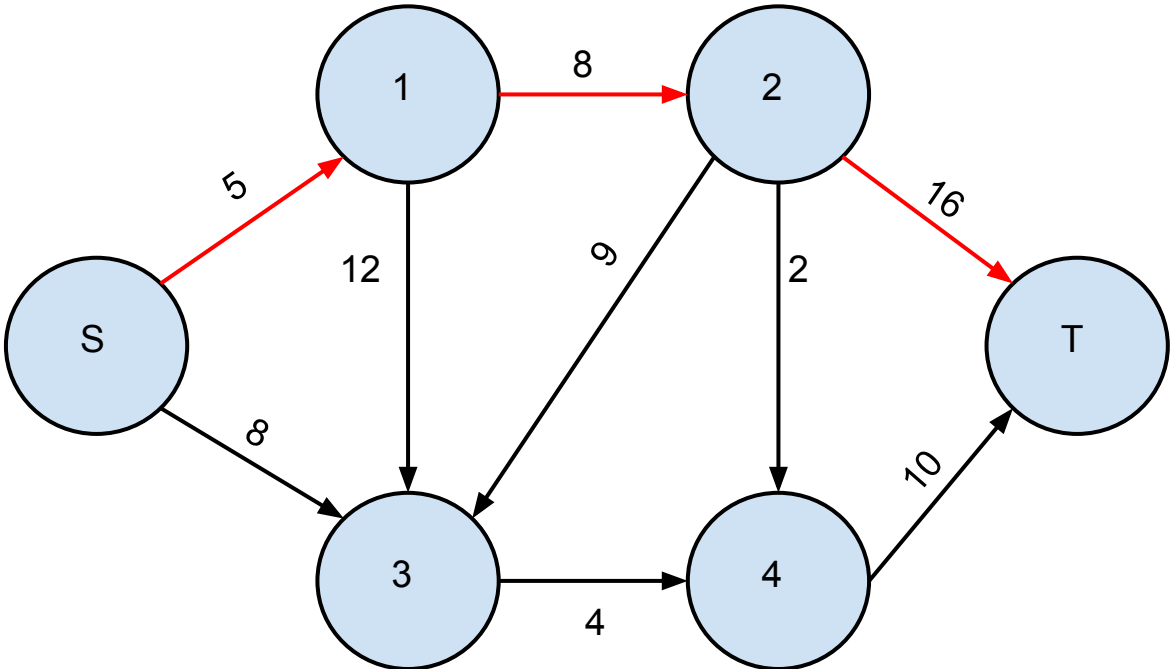
How does it all come together?



Combined Algorithm

Step 1:

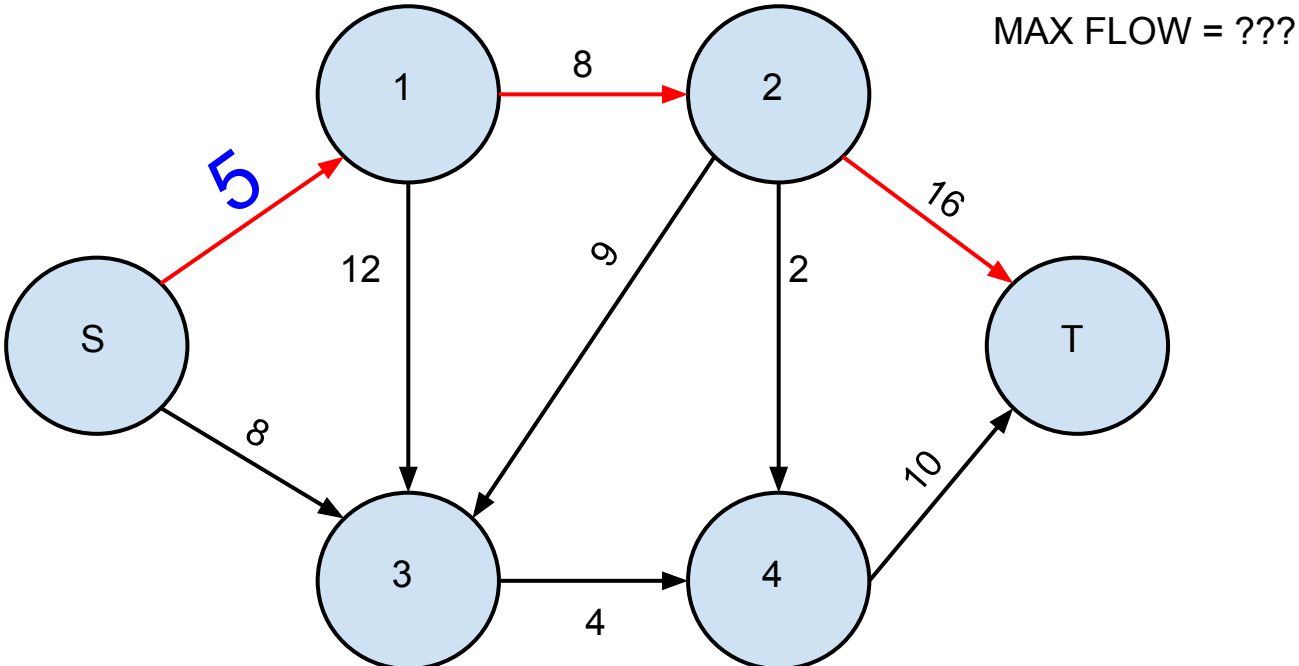
- BFS to find Shortest Path



Combined Algorithm

Step 2:

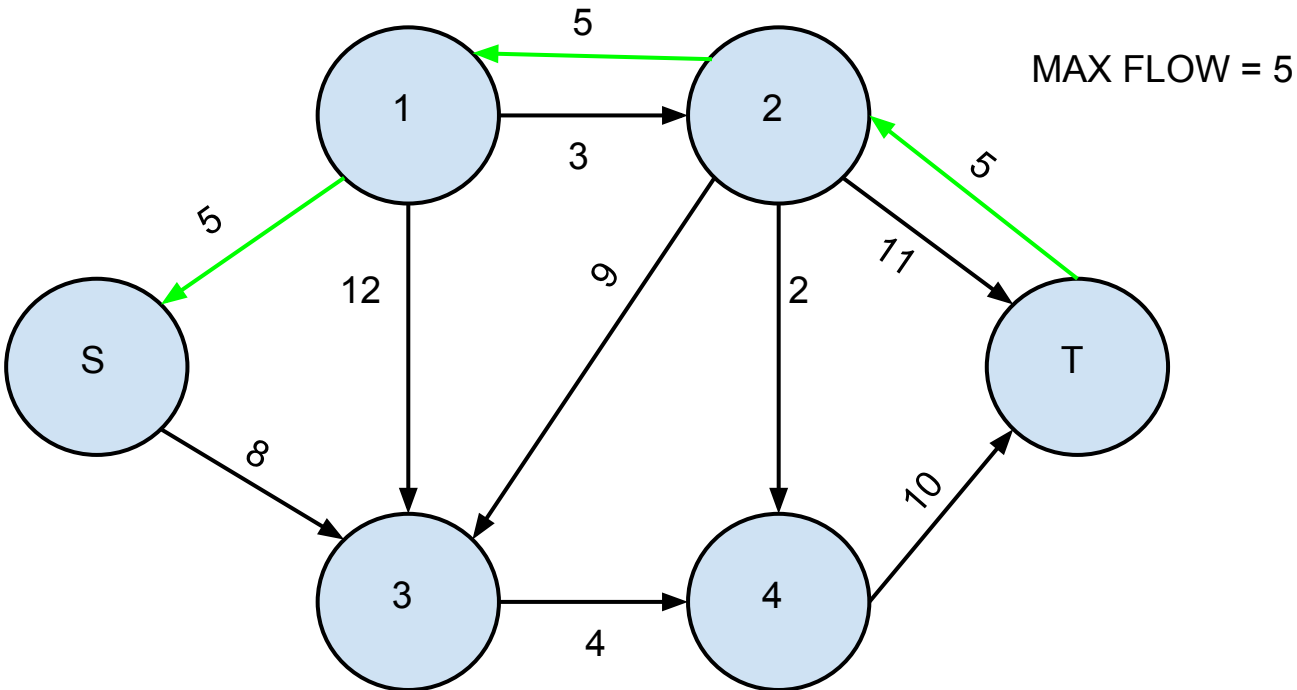
- Find the maximum flow (smallest edge weight) in the shortest path.



Combined Algorithm

Step 3:

- Add the max flow for the path to the max flow for the entire graph, and adjust edge weights using the maximum flow of the path resulting in a residual graph.



Combined Algorithm

Repeat the process until there are no paths from S to T in the residual graph

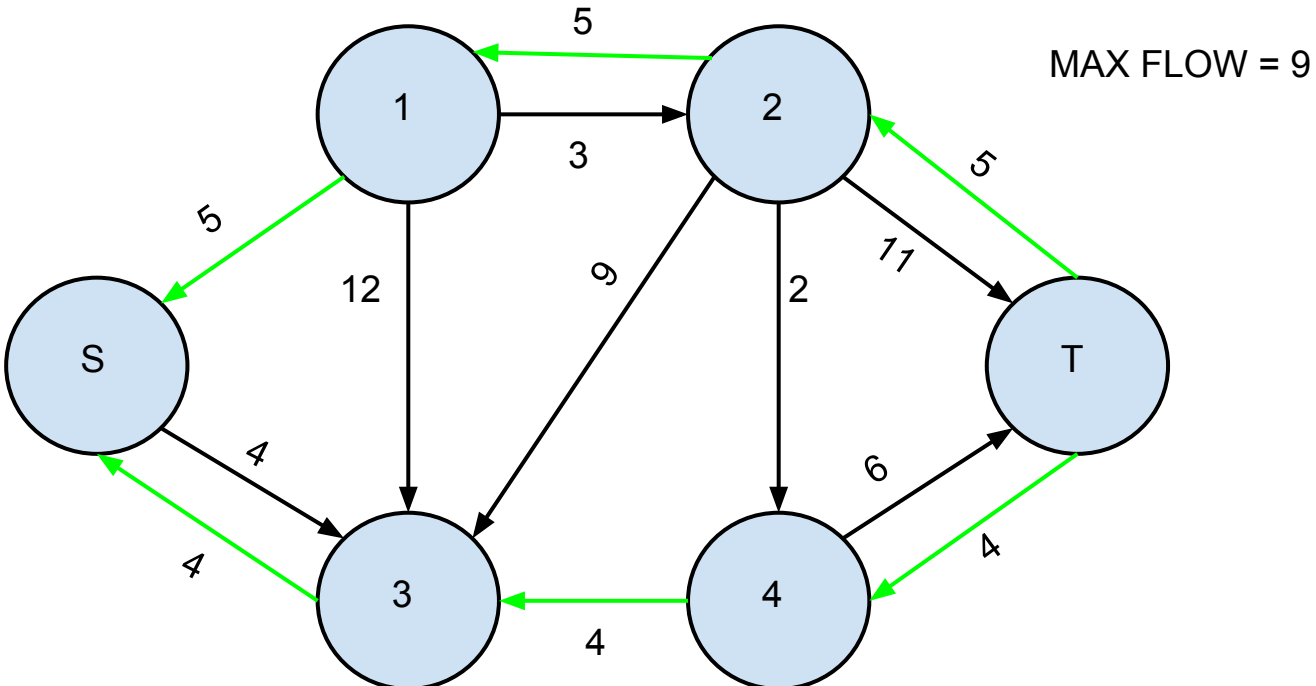


Image Segmentation

- GOAL
  - Separate image pixel elements that represent foreground content from those that represent background content.

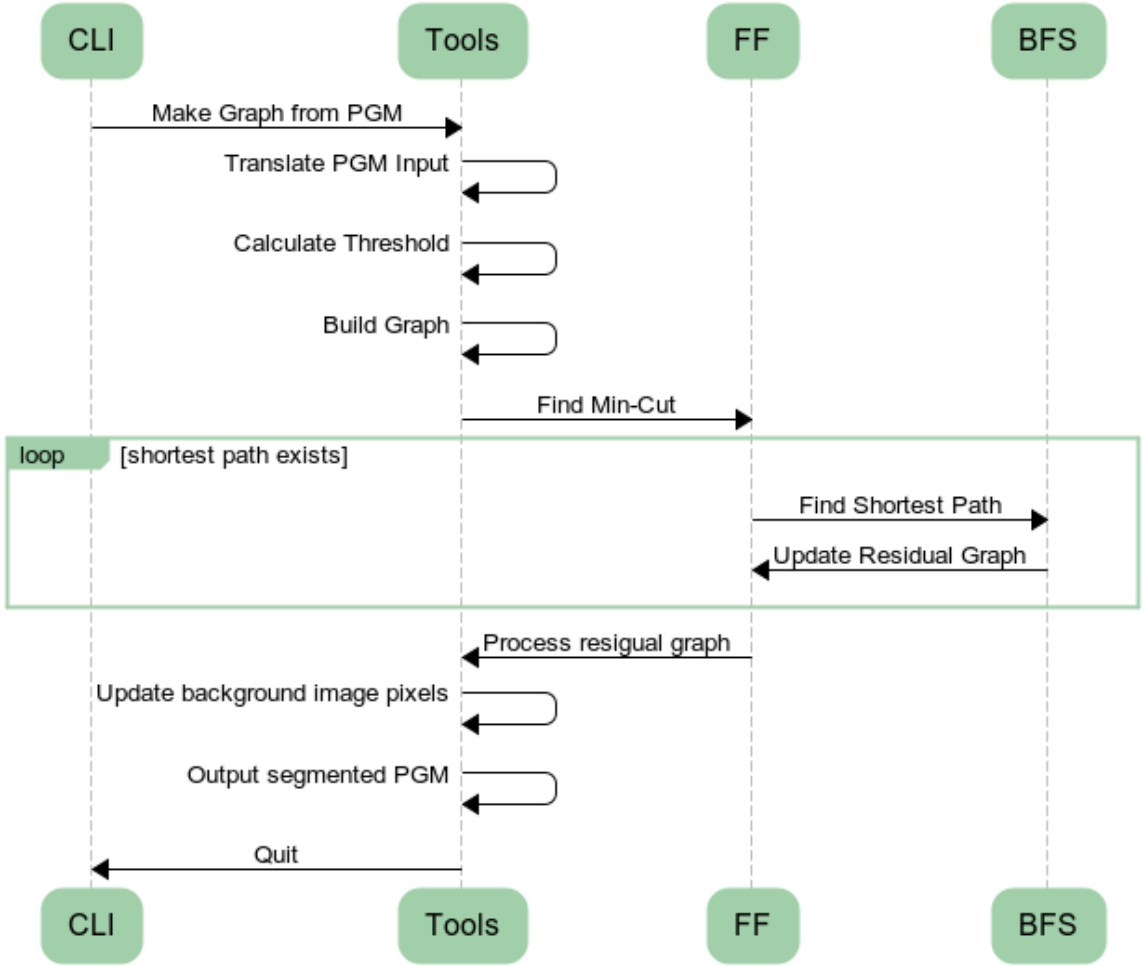
Image Segmentation - Setup

- After reading in PGM file (Optimization)
  - Calculate a threshold value
    - Threshold = | Max Pixel - | Sum of nodes / num of nodes |
  - Calculate edge penalties (S = 0, T = Max)
    - Penalty = Max Pixel - | node - node neighbor |
  - Build Adjacency List
    - Only connect nodes with edges where Threshold > Penalty

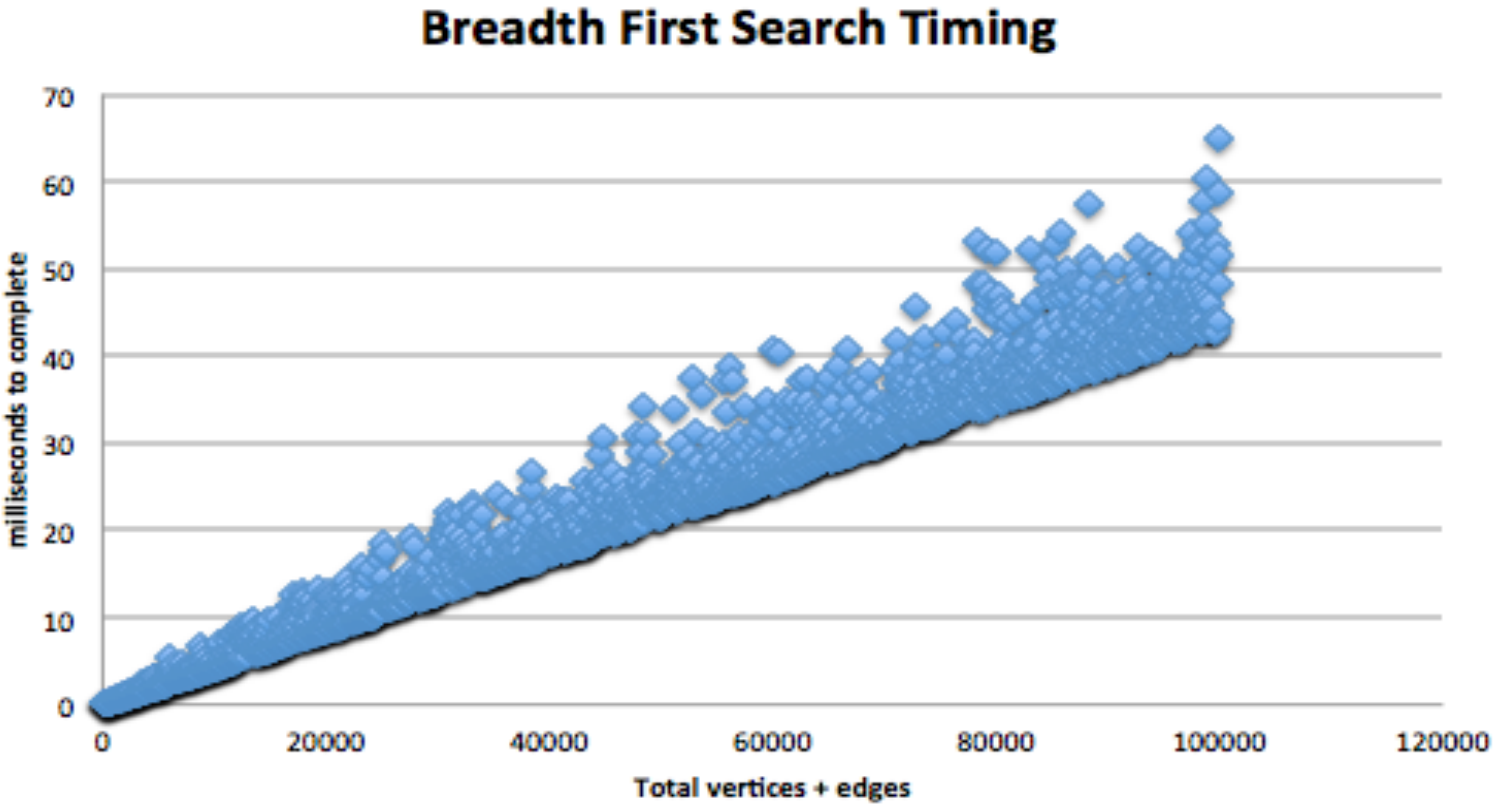
Image Segmentation - Process

- Run Ford-Fulkerson until no path exists between S(source) and T(sink)
- Use residual graph to determine which pixels are in the background and foreground
- Update original image to show all background pixels as white (Max value)

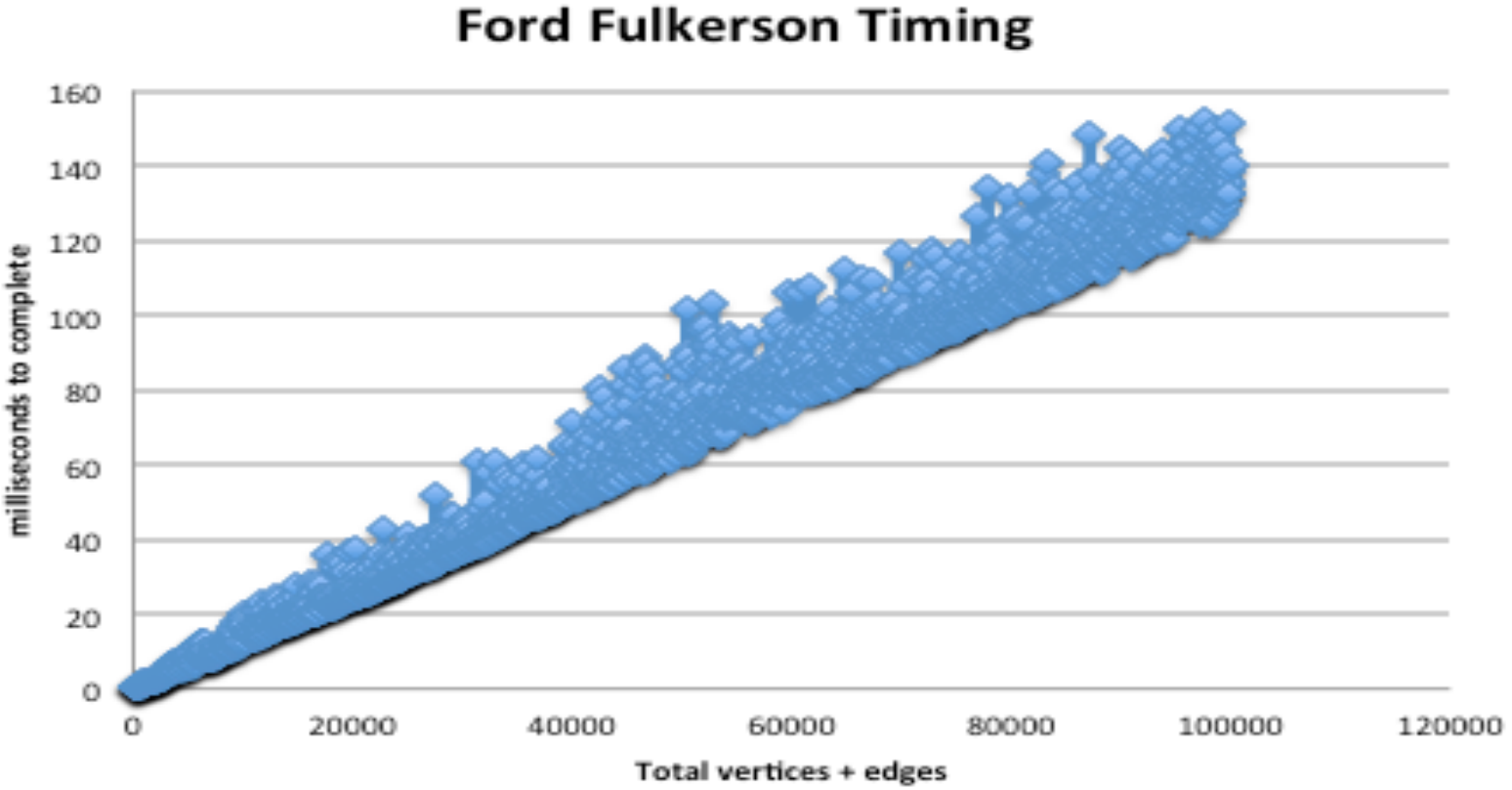
Image Segmentation



Results


Breadth First Search Timing
milliseconds to complete
70
60
50
40
30
20
10
0
0
20000
40000
60000
80000
100000
120000
Total vertices + edges

Results


Ford Fulkerson Timing
milliseconds to complete
160
140
120
100
80
60
40
20
0
0
20000
40000
60000
80000
100000
120000
Total vertices + edges

Results

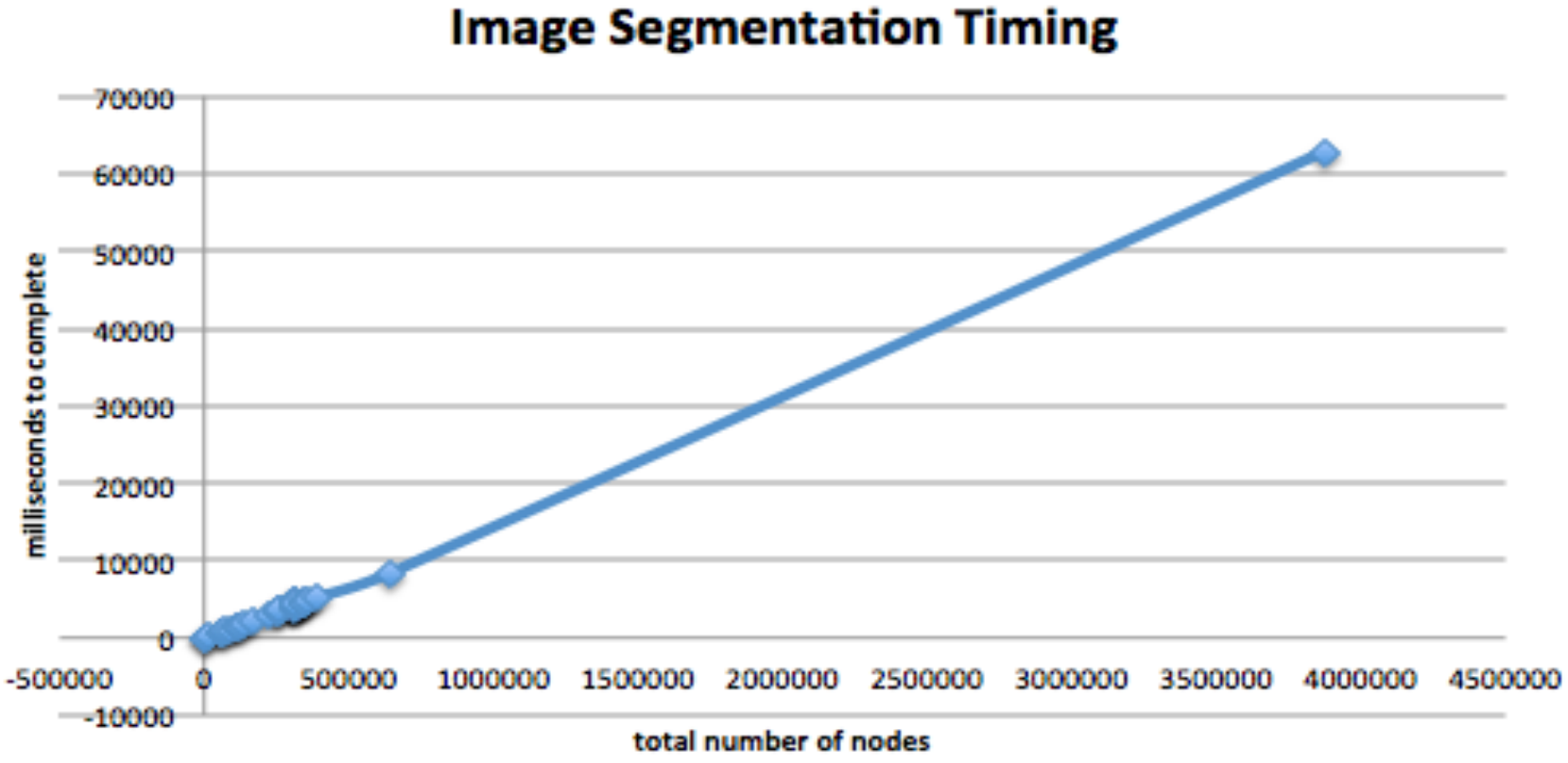


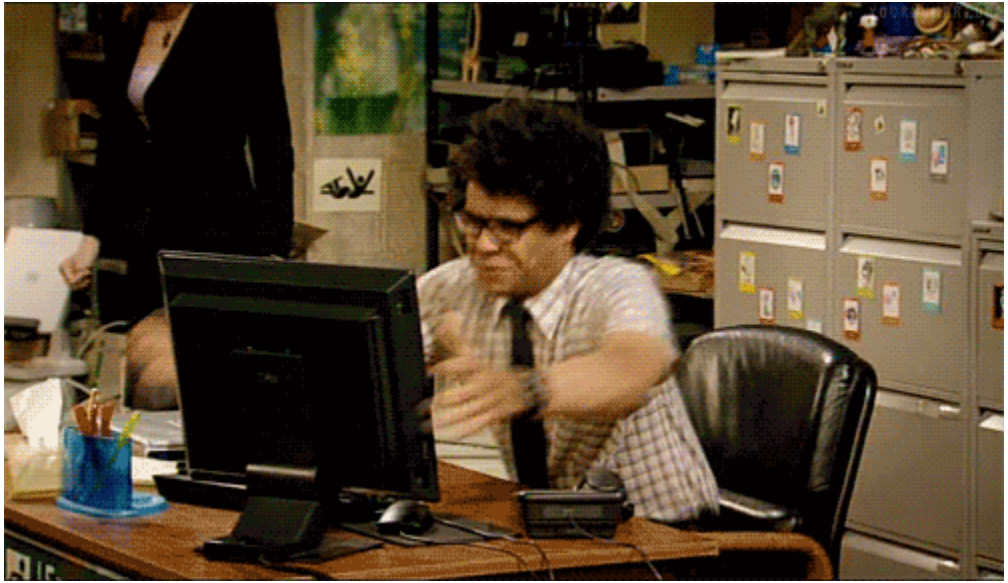
Image Segmentation Timing
70000
60000
50000
40000
30000
20000
10000
0
-10000
milliseconds to complete
-500000
0
500000
1000000
1500000
2000000
2500000
3000000
3500000
4000000
4500000
total number of nodes

Results

Conclusions

- Theoretical predictions are often worse than the empirical evidence, especially with small data sets
- Optimization goes a long way
- Depending on the application, a light-weight “good-enough” algorithm can work as a substitute

DEMO TIME



Q & A