December 6 – Maximum Flow
G Carl Evans
Origin of Maxflow Problem

Figure 7 — Traffic pattern: entire network available

Legend:
- — International boundary
- Railway operating division
- Capacity: 12 each way per day.
- Required flow of 3 per day toward destinations in direction of arrows, with equivalent number of returning trains in opposite direction.

All capacities in 1000's of tons each way per day

Origins: Divisions 2, 3, 6, 8 (Poland), 12, 52 (USSR), and Romania

Destinations: Divisions 3, 6, 8 (Poland), 9 (Czechoslovakia), and 2, 5 (Austria)

Alternative destinations: Germany or East Germany

Note 11C of Division 9, Poland
We are given as input graph $G_1$.

We create two new graphs: a flow graph $F$ and a residual graph $R$.

**Problem 1.**

The algorithm works by selecting paths from the residual graph $R$. The first path selected is $A \rightarrow B \rightarrow C \rightarrow F$ in graph $R$. This path’s flow capacity is 3. What do you think determines the flow capacity?
The algorithm uses the path to modify graphs $F$ and $R$. Here is the result.

**Graph $F$**

**Graph $R$**

**Problem 2.**
Examine the new versions of $F$ and $R$ above. What is being done with the path selected from $R$ to modify these graphs?

**Problem 3.**
The next path selected was $A \rightarrow D \rightarrow E \rightarrow F$ in graph $R$. What is the flow capacity of that path?
The resulting working graphs are these:

**Graph F**

![Graph F Diagram]

**Graph R**

![Graph R Diagram]

**Problem 4.**
We select path $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F$. What is the flow capacity of that path?

**Problem 5.**
The paths selected always start from node $A$ and end with node $F$. What is different about these nodes compared to the others?
Here are the final working graphs $F$ and $R$.

Graph $F$

Graph $R$

Problem 6.
At this point, the algorithm is finished. How can we know the algorithm is done by examining graph $R$?

Problem 7.
For nodes $B$, $C$, $D$, and $E$, what is the relationship between the in-flows and the out-flows? Why does that relationship have to exist?
Here are the final working graphs $F$ and $R$.

**Graph $F$**

**Graph $R$**

**Problem 8.**
Using the final flow graph $F$ above, determine the maximum flow of graph $G1$.

**Problem 9.**
In graph $F$, the outflow of $A$ is equal to the inflow of $F$. Should that always be the case?
Here are the final working graphs $F$ and $R$.

**Problem 10.**

Node $A$ is called a *source node* and node $F$ is called a *sink node*. Would this technique work if there were multiple source and sink nodes? Why or why not?
Now we are going to look at a case that messes up the algorithm.

**Graph G2**

**Flow Graph**

**Residual Graph**

**Problem 11.**

The algorithm picks path $A \rightarrow B \rightarrow C \rightarrow D$. What is the capacity of that path?

**Problem 12.**

Update the flow and residual graphs as a result of selecting this path.
Problem 17.
What is the maximum network flow of $G_2$, according to the algorithm?

Problem 18.
Is this number correct? Why or why not? Examine $G_2$ to verify your answer.

Problem 19.
Suppose $G_2$ modeled a network of water pipes. What would happen on edge $B \rightarrow C$ in this situation? Would it change the total flow of $G_2$ if we deleted that edge?
We are going to modify the algorithm. Starting again with the previous graph, we make a new kind of residual graph. The dotted edges are added, and are legal edges to be traversed in the residual graph.

**Problem 20.**
Select path $A \rightarrow B \rightarrow C \rightarrow D$. What is the capacity of that path?
Flow Graph

Residual Graph

Now we select path $A \rightarrow C \rightarrow B \rightarrow D$.
Here are the updated flow and residual graphs:

Flow Graph

Residual Graph
Problem 22.
Select path $A \rightarrow B \rightarrow C \rightarrow D$. (Yes, we are repeating this path.) What are the resulting flow and residual graphs?
Problem 23.
Now we select path $A \to C \to B \to D$.
What are the updated flow and residual graphs?

Flow Graph
![Flow Graph Diagram]

Residual Graph
![Residual Graph Diagram]

Problem 24.
At this point, the algorithm should be done. Is the final network flow accurate now?
Ford Fulkerson Requirements and Runtime
Ford and Fulkerson's augmenting-path algorithm

Ford and Fulkerson's proof of the Max-flow Min-cut Theorem immediately suggests an algorithm to compute maximum flows: Starting with the zero flow, repeatedly augment the flow along any path from \( s \) to \( t \) in the residual graph, until there is no such path.

This algorithm has an important but straightforward corollary: Integrality Theorem. If all capacities in a flow network are integers, then there is a maximum flow such that the flow through every edge is an integer.

Proof: We argue by induction that after each iteration of the augmenting path algorithm, all flow values and all residual capacities are integers.

- Before the first iteration, all flow values are 0 (which is an integer), and all residual capacities are the original capacities, which are integers by definition.
- In each later iteration, the induction hypothesis implies that the capacity of the augmenting path is an integer, so augmenting changes the flow on each edge, and therefore the residual capacity of each edge, by an integer. In particular, each iteration of the augmenting path algorithm increases the value of the flow by a positive integer. It follows that the algorithm eventually halts and returns a maximum flow.

If every edge capacity is an integer, then conservatively, the Ford-Fulkerson algorithm halts after at most \(|f^*|\) iterations, where \( f^* \) is the actual maximum flow. In each iteration, we can build the residual graph \( G_f \) and perform a depth-first-search to find an augmenting path in \( O(E) \) time. Thus, in this setting, the algorithm runs in \( O(E|f^*|) \) time in the worst case.

Jack Edmonds and Richard Karp observed that this running time analysis is essentially tight. Consider the \( n \)-node network in Figure 10.7, where \( X \) is some large integer. The maximum flow in this network is clearly \( 2X \). However, Ford-Fulkerson might alternate between pushing one unit of flow along the augmenting path \( s \rightarrow u \rightarrow t \) and then pushing one unit of flow along the augmenting path \( s \rightarrow v \rightarrow t \), leading to a running time of \( \mathcal{O}(X) = \Omega(|f^*|) \).

**Figure 10.7.** Edmonds and Karp's bad example for the Ford-Fulkerson algorithm.

Image from https://jeffe.cs.illinois.edu/teaching/algorithms/book/10-maxflow.pdf
Edmonds and Karp’s Algorithms