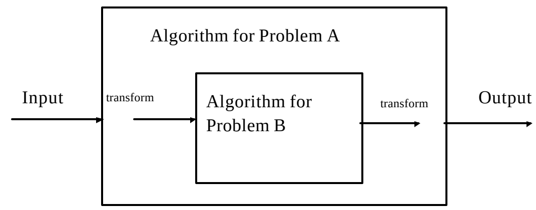
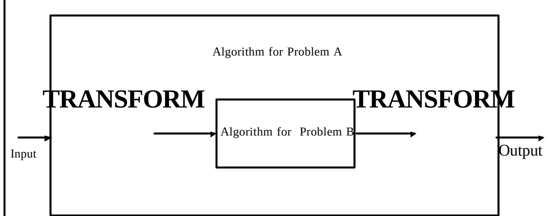
Algorithm Design Techniques

- Induction
- Divide and Conquer
- Dynamic Programming
- Greedy
- **Reduction**

Reduction: A ∝ B



Reduction: A ∝ B



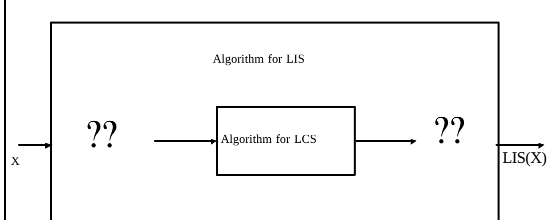
Longest Increasing Subsequence ∝ Longest Common Subsequence

- **Longest Increasing Subsequence**
  - Input: Sequence of integers X
  - Output: Longest subsequence $y_1, y_2, \ldots, y_k$ of X such that $y_i < y_{i+1}$ for $i=1,\ldots,k-1$
- Example
  - Input: 2, 4, 3, 1, 9, 7 ,5,6
  - Output: 2,3,5,6

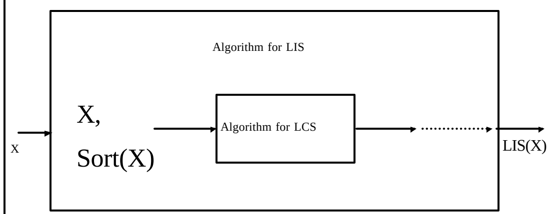
Longest Increasing Subsequence ∝ **Longest Common Subsequence**

- Longest Common Subsequence
  - Input: Two sequences of integers X and Y
  - Output: Longest subsequence of X that is also a subsequence of Y
- Example
  - Input: 1,5,3,2,7,4,12 and 5,12,4,12,2,12
  - Output: 5,2,12

Reduction: LIS ∝ LCS



Reduction: LIS ∝ LCS



Correctness

- Claim: Let X be a sequence of integers and let Y be a subsequence of X. Y is an increasing subsequence of X if and only if it is also a subsequence of Sort(X).

Correctness

Claim: Let X be a sequence of integers and let Y be a subsequence of X. Y is an increasing subsequence of X if and only if it is also a subsequence of Sort(X).

Thus any longest common subsequence of X and Sort(X) is a longest increasing sequence of X

Efficiency

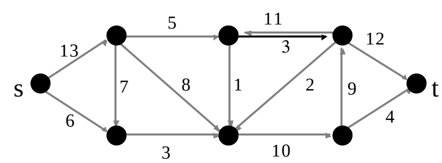
- $T_{LIS}(n) = O(n \lg(n)) + T_{LCS}(2n)$

Some reductions we've seen

- Sorting ∝ Find-max
- General Selection ∝ Find-median
- All pairs shortest path ∝ all pairs all restricted shortest paths

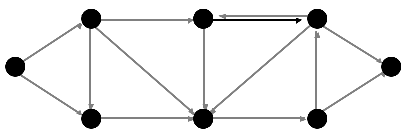
Max Flow in a Network

- Input: Flow Network



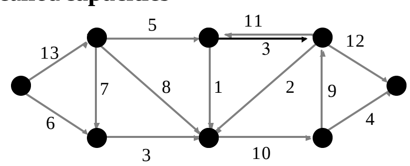
Flow Network

- Directed graph



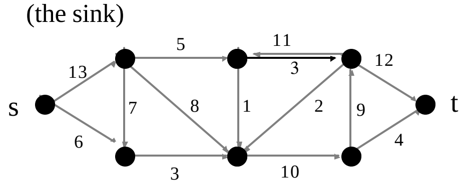
Flow Network

- Directed graph with positive edge weights called **capacities**



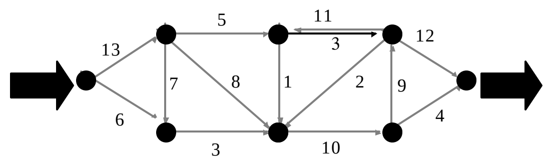
Flow Network

- Directed graph with positive edge capacities and two special vertices s (the source) and t (the sink)



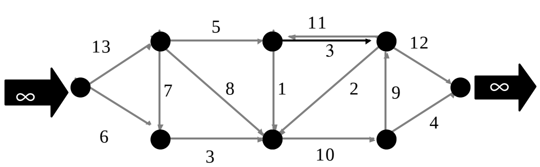
Max Flow in a Network

- Output: Maximum flow that can be pumped through the network from s to t.



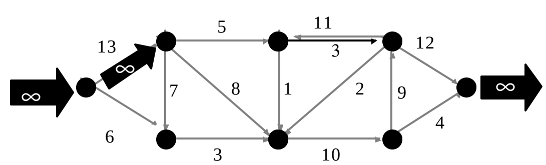
The Rules

0. The source has infinite input capacity. The sink has infinite output capacity.



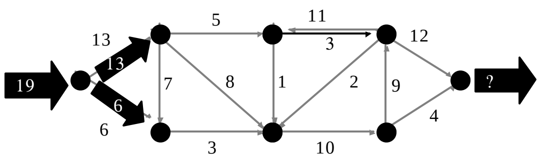
The Rules

1. The total flow into a node must equal the total flow out of a node.



The Rules

2. The flow along an edge cannot exceed its capacity. (No edge means 0 capacity.)

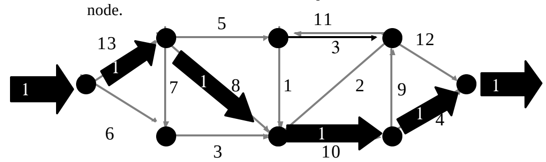


CS140 25-19

What is a feasible flow?

Any flow that satisfies the rules.

1. The flow along an edge cannot exceed its capacity
2. The total flow into a node must equal the total flow out of a node.



CS140 25-20

Conservation of Flow

- Flow in = Flow out (Rule 2)
- Flow into s = Flow out of t

CS140 25-21

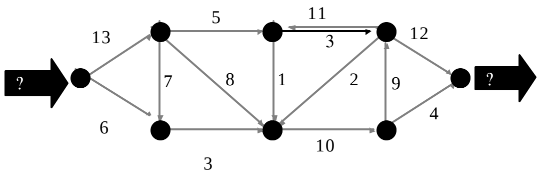
Network flow

- Flow into s = Flow out of t = Network flow

CS140 25-22

What is the max flow?

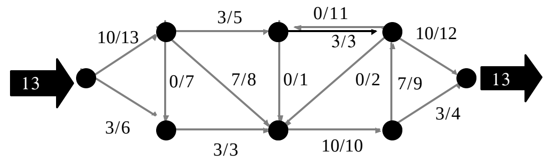
1. The flow along an edge cannot exceed its capacity
2. The total flow into a node must equal the total flow out of a node.



CS140 25-23

MAX FLOW?

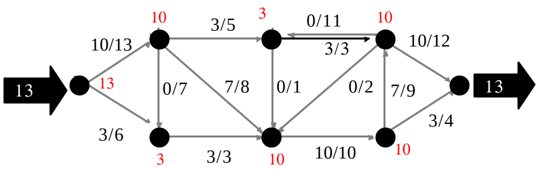
1. **The flow along an edge cannot exceed its capacity**
2. The total flow into a node must equal the total flow out of a node.



CS140 25-24

MAX FLOW?

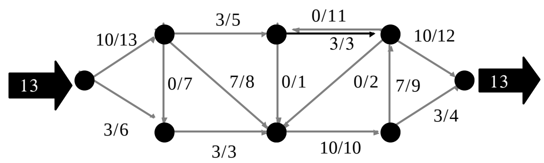
1. The flow along an edge cannot exceed its capacity
2. **The total flow into a node must equal the total flow out of a node.**



CS140 25-25

Feasible Flow! But is it a Max Flow?

1. The flow along an edge cannot exceed its capacity
2. The total flow into a node must equal the total flow out of a node.



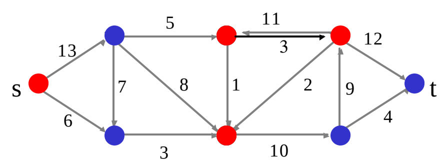
CS140 25-26

Cut of a Flow Network

- Cut (of a flow network) is a partition of the vertices into two sets R and B such that s ε R and t ε B

CS140 25-27

Example of a cut



CS140 25-28

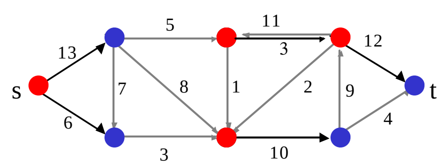
Capacity of a cut

- The capacity of a cut (R,B) of a network is the sum of the capacities of the edges that go from a R vertex to a B vertex.

CS140 25-29

Example of cut capacity

(R,B) has capacity 41



CS140 25-30

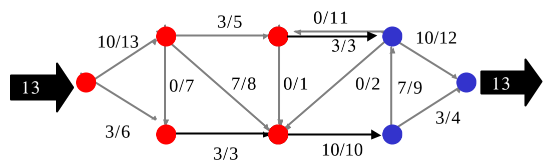
Max Flow-Min Cut Theorem

- The maximum flow of a network is equal to the minimum capacity of any cut in the network.

CS140 25-31

Feasible Flow!
But is it a Max Flow?

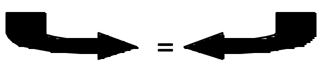
Here is a matching cut! But is it a min-cut?



CS140 25-32

Max Flow-Min Cut Theorem

- The maximum flow of a network is equal to the minimum capacity of any cut in the network.
- Flow $\leq$ Max flow = Min cut $\leq$ Cut



CS140 25-33

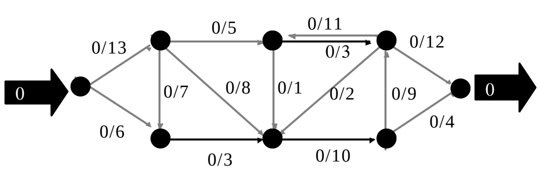
Network Flow
What is ahead?

- Algorithms for finding maximum flow in a network
- Reducing problems to the max flow problem

CS140 25-34

Augmenting Path Algorithm

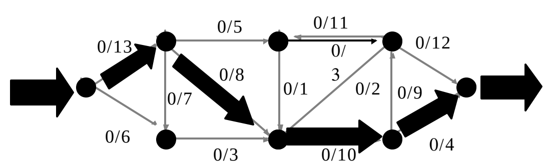
- Find an augmenting path



CS140 25-35

Augmenting path

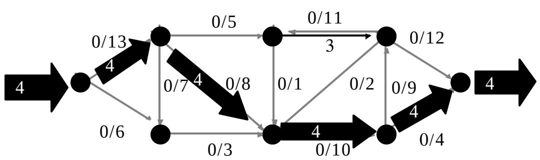
Find an augment path: An s => t path in which the capacity of each edge exceeds its flow.



CS140 25-36

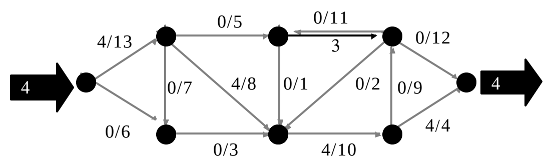
Augmenting path

1. Find an augment path
2. Push as much flow through the path as possible.



Repeat

1. Find an augment path
2. Push as much flow through the path as possible.



Augmenting paths

- Does this work?
- Can we do it efficiently?