Week 10 - Friday

COMP 2100

Last time

- What did we talk about last time?
- Matching practice
- Stable marriage
- Euler paths and cycles

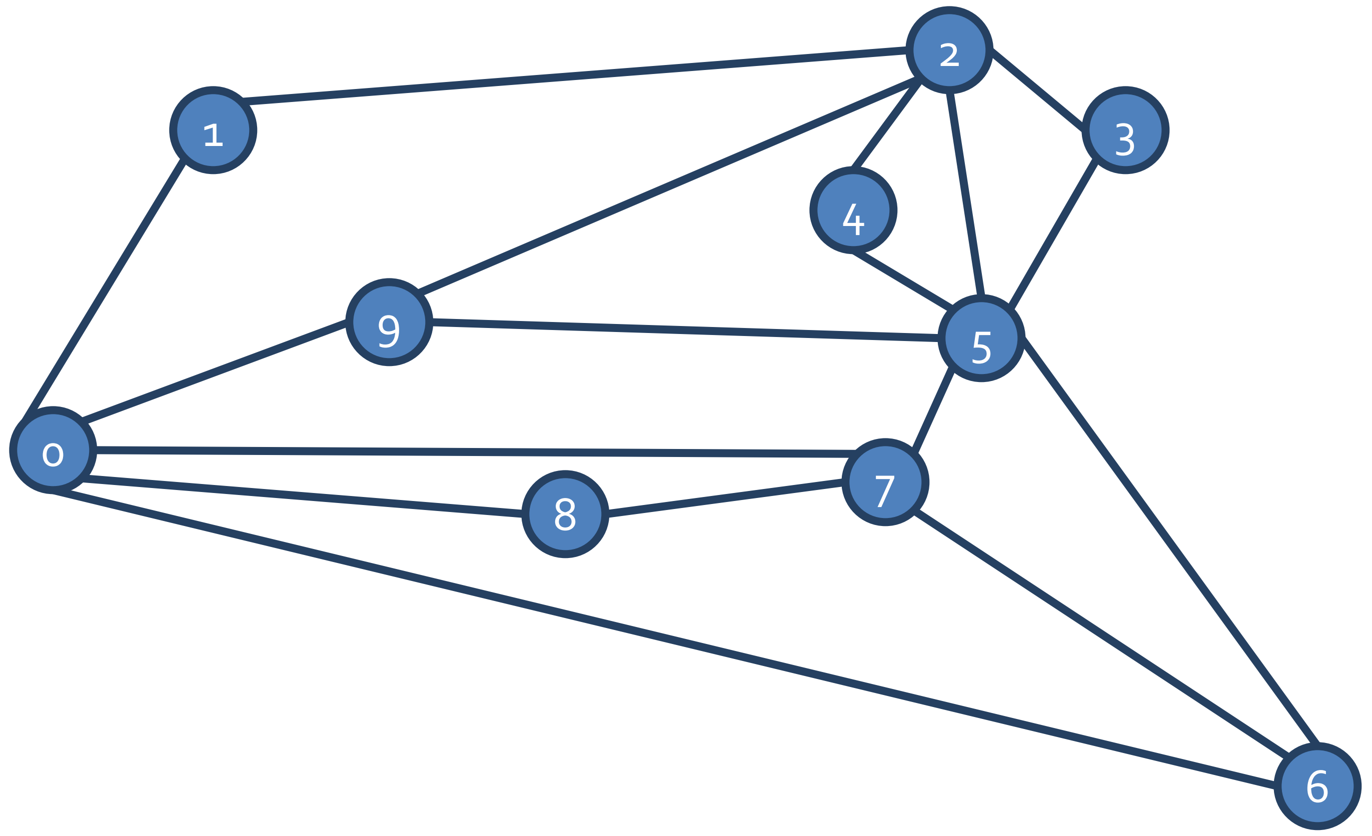
Questions?

Project 3

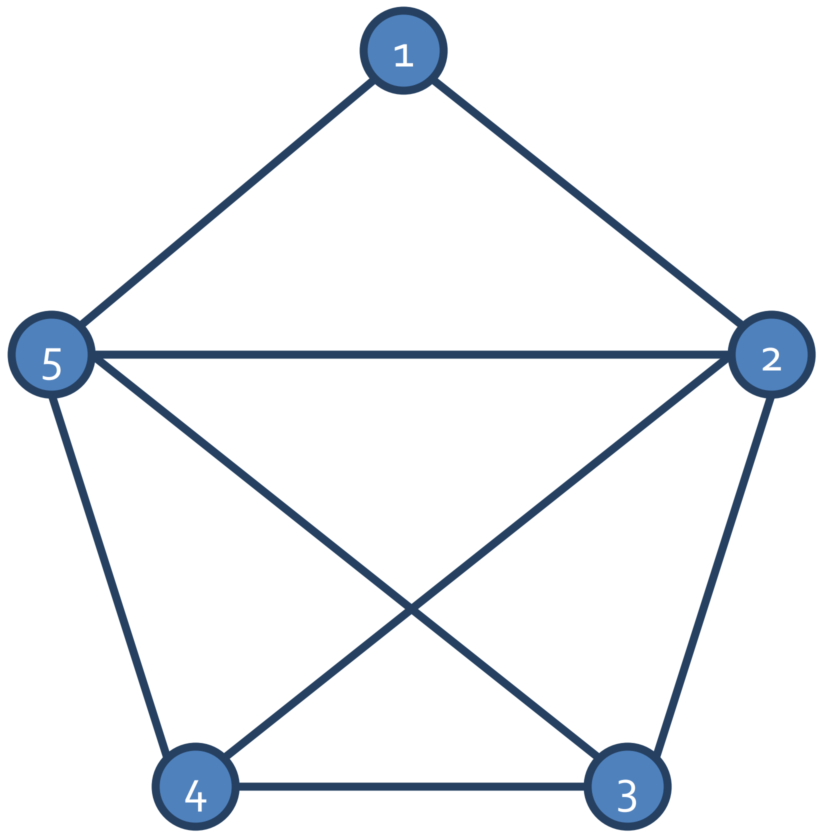
Assignment 5

Euler Practice

Is there an Euler path or cycle?



Is there an Euler path or cycle?



Network Flow

Flow networks

- A **flow network** is a weighted, directed graph with positive edge weights
  - Think of the weights as **capacities**, representing the maximum units that can flow across an edge
- An ***st*-flow network** has a **source *s*** (where everything comes from) and a **sink *t*** (where everything goes to)

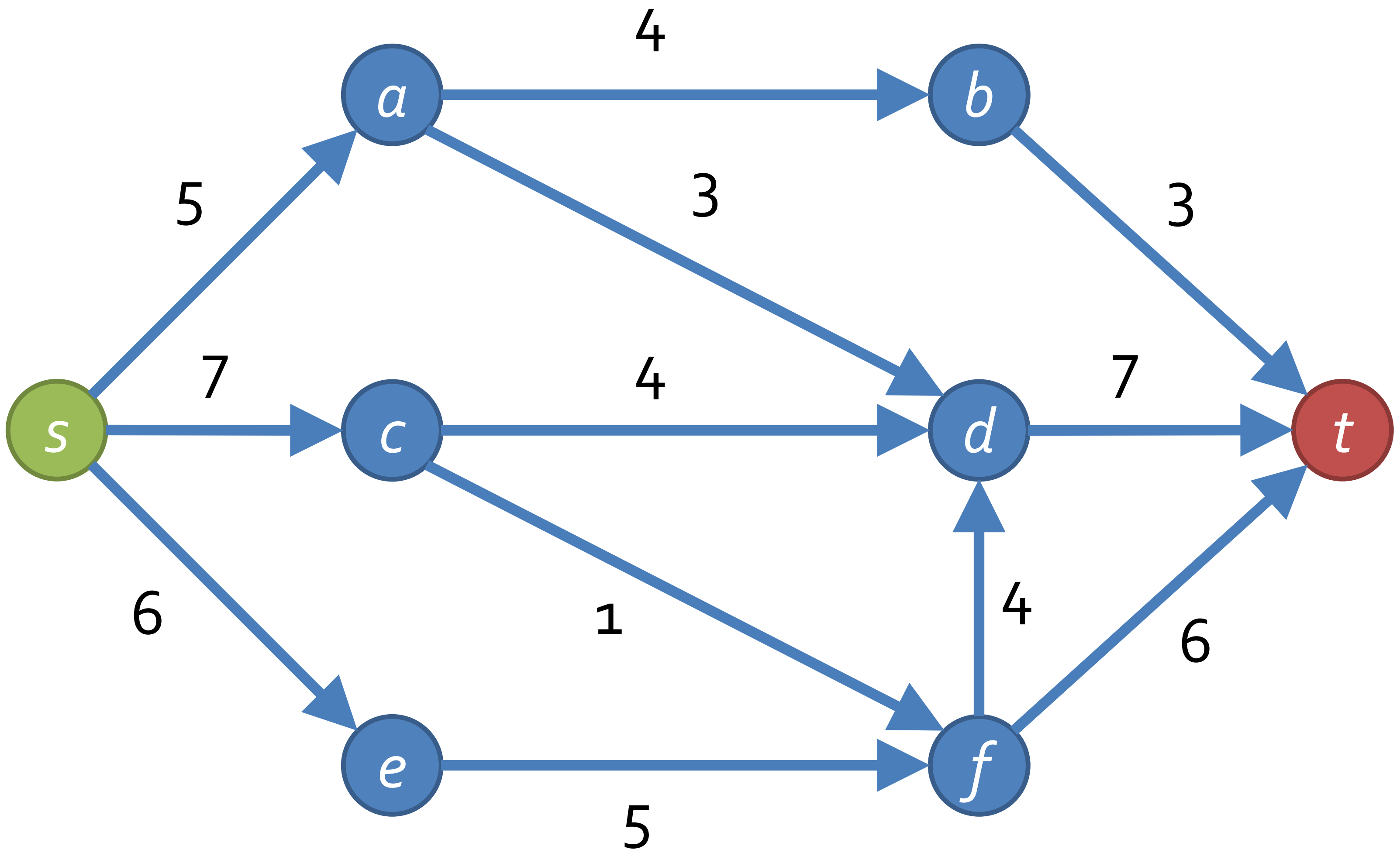
Applications of flow problems

- Oil flowing from a start to a destination
- Airline crews needed to man aircraft, moving from city to city
- Goods being produced by factories and consumed by cities, with roads that can accommodate a certain amount of traffic

Maximum flow

- A common flow problem is to find the **maximum flow**
- A maximum flow is a non-negative amount of flow on each edge such that:
  - The maximum amount of flow gets from ***s*** to ***t***
  - No edge has more flow than its capacity
  - The flow going into every node (except ***s*** and ***t***) is equal to the flow going out

Flow network



Augmenting path

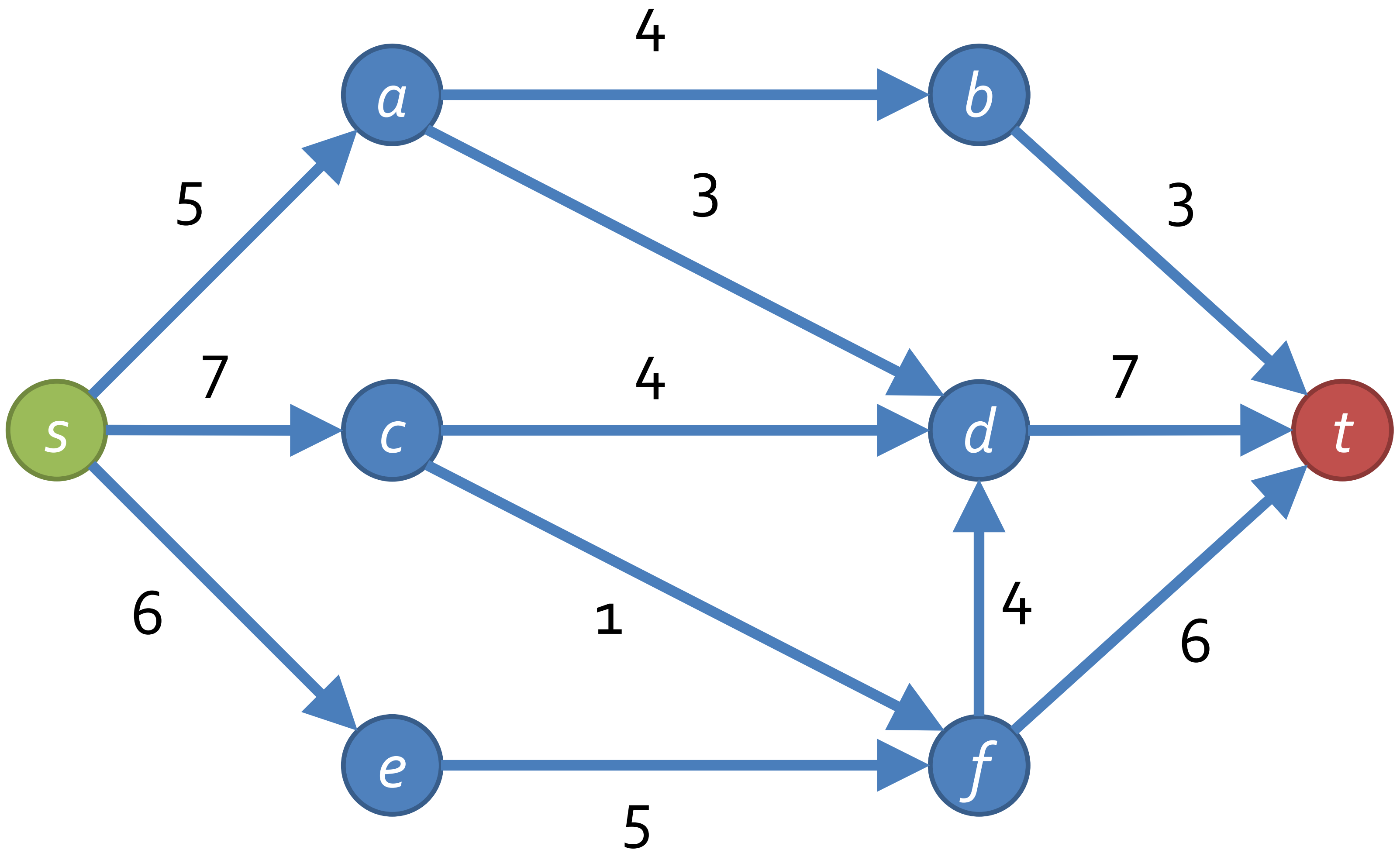
- When we were talking about matching, we mentioned **augmenting paths**
- Augmenting paths in flows are a little different
- A flow augmenting path:
  - Starts at ***s*** and ends at ***t***
  - May cross some edges in the direction of the edge (forward edges)
  - May cross some edges in the opposite direction (backwards edges)
  - Increases the flow by the minimum of the unused capacity in the forward edges or the maximum of the flow in the backwards edges

Ford-Fulkerson algorithm

- Ford-Fulkerson is a family of algorithms for finding the maximum flow

1. Start with zero flow on all edges
2. Find an augmenting path (increasing flow on forward edges and decreasing flow on backwards edges)
3. If you can still find an augmenting path, go back to Step 2

Find a max flow



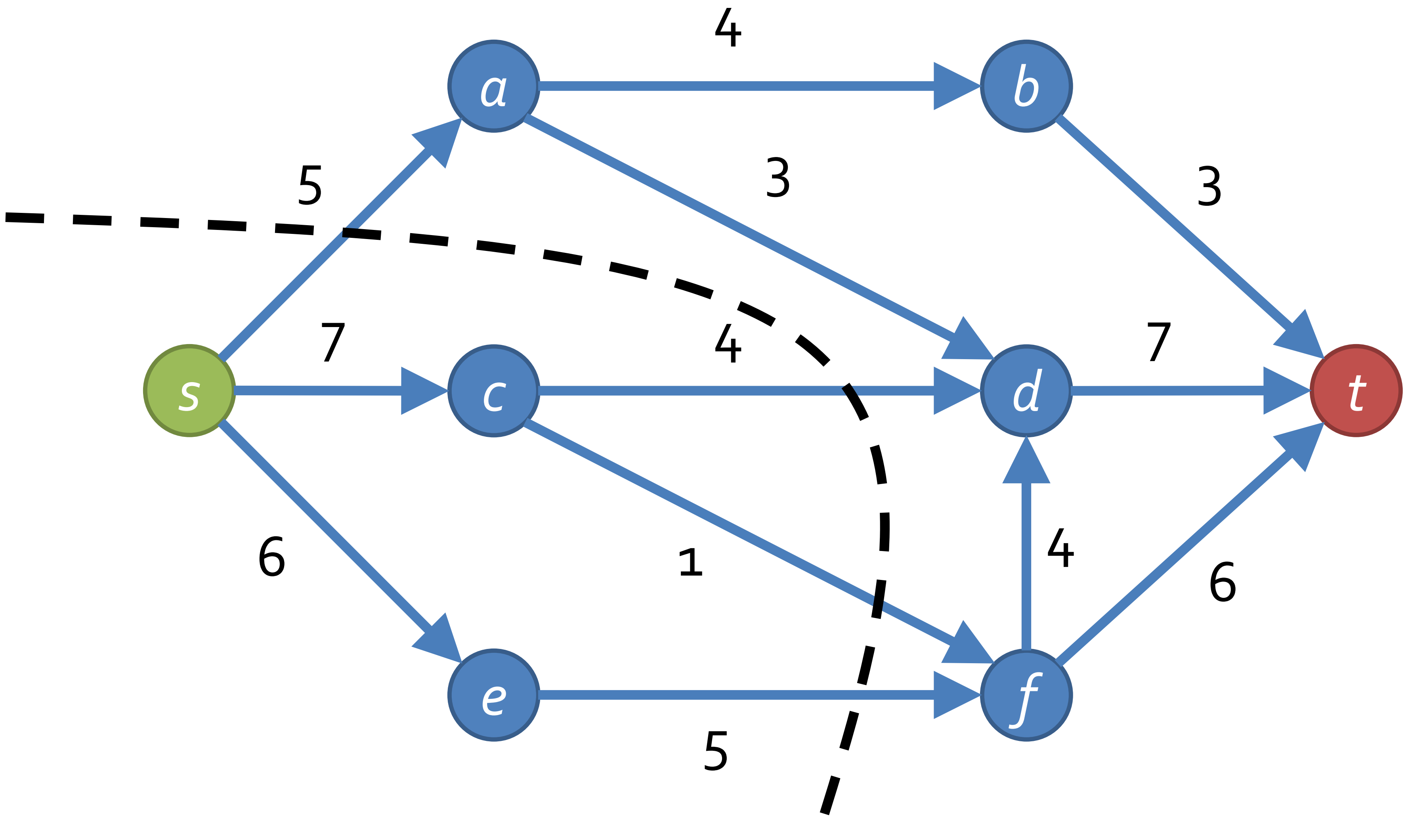
Cuts

- A **cut** in a graph partitions the graph into two disjoint sets
- An ***st*-cut** is a cut such that ***s*** is in one partition and ***t*** is in the other
- Think of it as a line that slices through the edges, putting ***s*** on one side and ***t*** on the other
- The **capacity of a cut** is the sum of the capacities of the edges that the cut divides

Maxflow-mincut theorem

- The smallest capacity ***st***-cut you can make has the same capacity as the largest possible ***st***-flow
- Intuitively, it's like that cut is a set of edges that most constricts the flow from ***s*** to ***t***

Minimum *st*-cut



Running time of Ford-Fulkerson

- Our definition of Ford-Fulkerson didn't say how you pick the augmenting path
- At worst, it could take O(|***E***|***f***), where |***E***| is the number of edges in the graph and ***f*** is the maximum flow
  - That could be terrible if ***f*** has a large numerical value
- Edmonds-Karp is a variation of Ford-Fulkerson that uses a breadth-first search to find a shortest augmenting path
  - It runs in O(|***V***||***E***|^2)

B-trees

Multiway trees

- Binary trees are great
- However, only two splits means that you have a height of $\log_2 n$ when you want to store n things
  - If n = 1,000,000, $\log_2 n$ = 20
- What if depth was expensive? Could we have say, 10 splits?
  - If n = 1,000,000, $\log_{10} n$ = 6

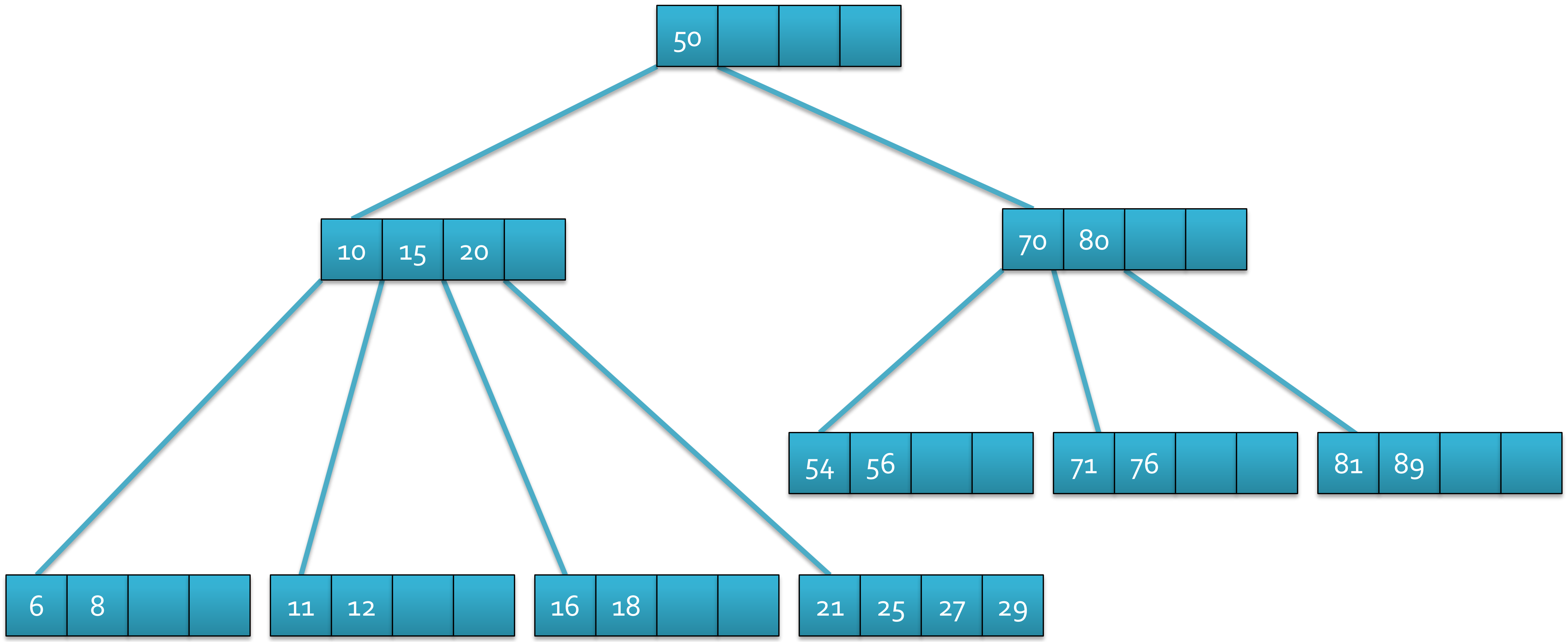
When would we need such a thing?

- **Answer:** When the tree is in secondary storage
- Each read of a block from disk storage is slow
  - We want to get a whole node at once
  - Each node will give us information about lots of child nodes
  - We don't have to make many decisions to get to the node we want

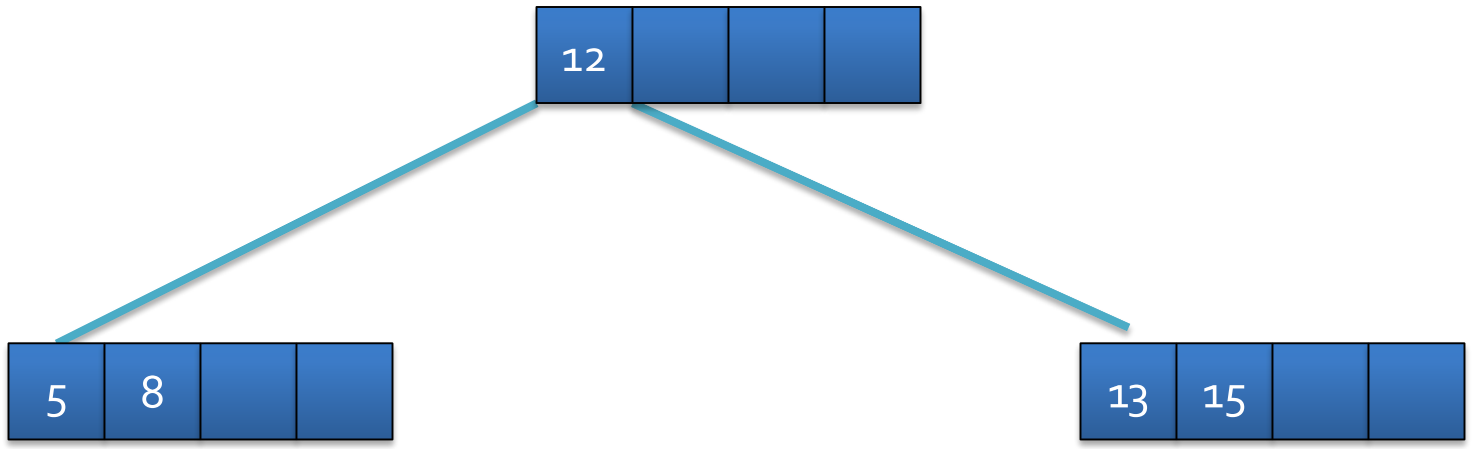
B-tree definition

- A B-tree of order ***m*** has the following properties:
  1. The root has at least two subtrees unless it is a leaf
  2. Each nonroot and each nonleaf node holds ***k*** keys and ***k*** + 1 pointers to subtrees where $m/2 \leq k \leq m$
  3. Each leaf node holds ***k*** keys where $m/2 \leq k \leq m$
  4. **All leaves are on the same level**

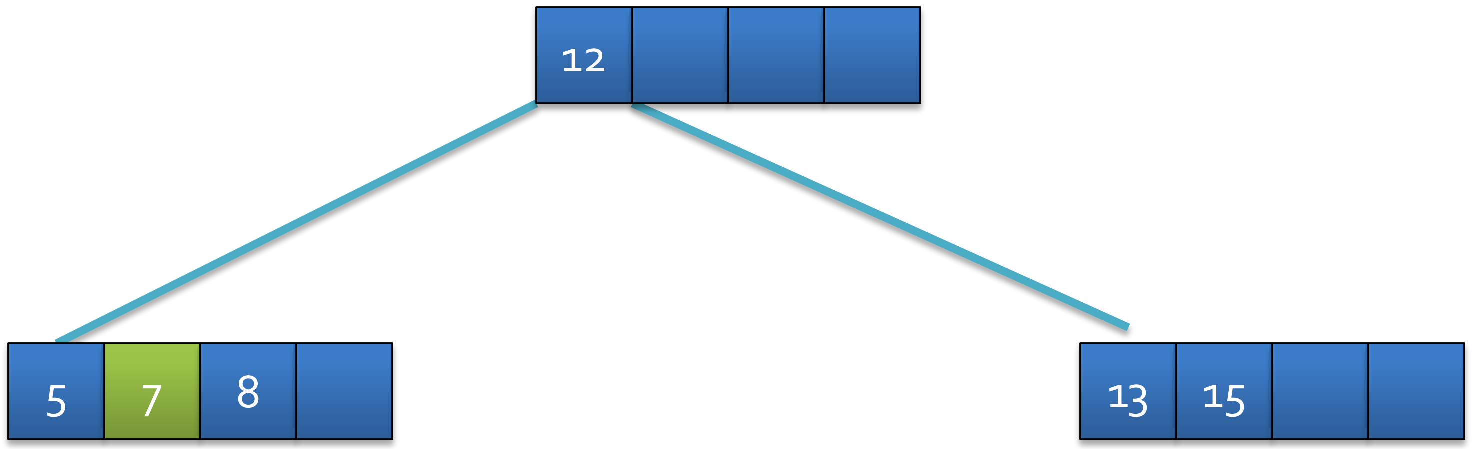
B-tree of order 4



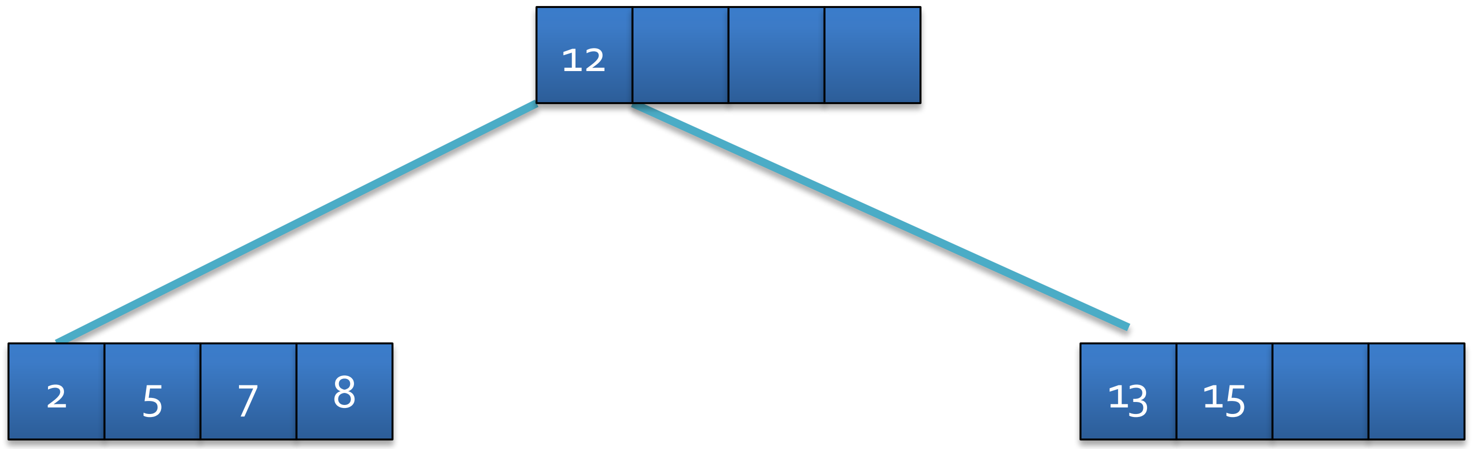
Easy add



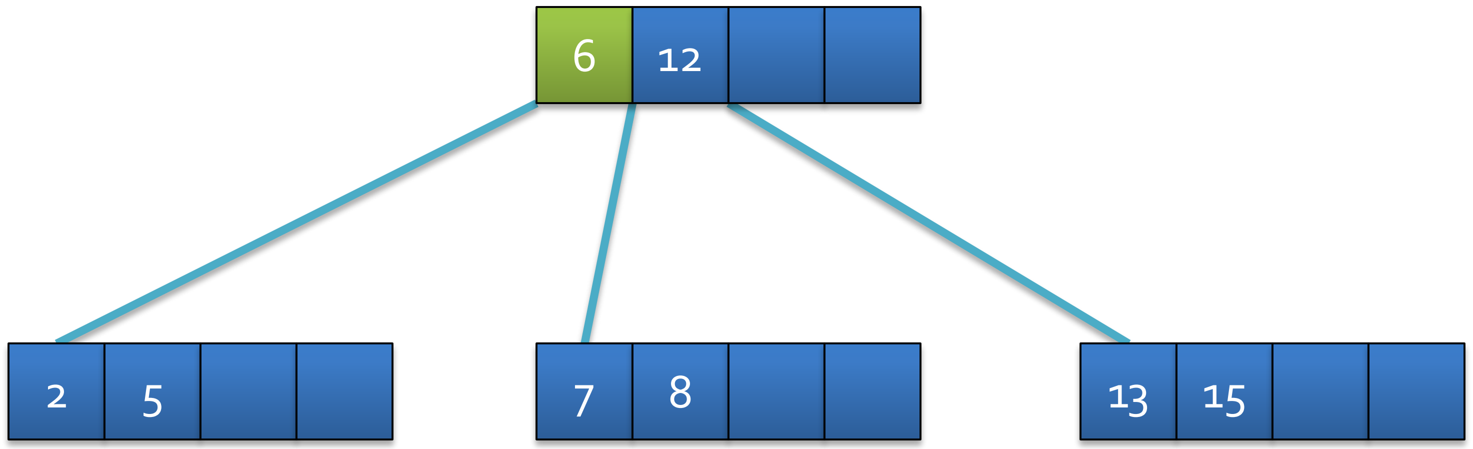
Insert 7



Tougher add



Insert 6



B-tree practice

- Insert the following numbers:
  - 86 69 81 15 100 94 8 27 56 68 92 89 38 53 88

Upcoming

Next time…

- Finish B-trees
- Intractability

Reminders

- Keep working on Project 3
  - **Due next Friday!**
- Finish Assignment 5
  - **Due tonight by midnight!**
- Read section 6.4